Cooperation

Huijink, Sybren

Publication date:
2016

Document Version
Publisher's PDF, also known as Version of record

Link to publication in Tilburg University Research Portal

Citation for published version (APA):
Huijink, S. (2016). *Cooperation: Vehicle routing and outsourcing, games and nucleoni*. [Doctoral Thesis, Tilburg University]. CentER, Center for Economic Research.

Cooperation: Vehicle Routing and Outsourcing, Games and Nucleoni

Cooperation: Vehicle Routing and Outsourcing, Games and Nucleoni

Proefschrift

ter verkrijging van de graad van doctor aan Tilburg University op gezag van de rector magnificus, prof. dr. E.H.L. Aarts, in het openbaar te verdedigen ten overstaan van een door het college voor promoties aangewezen commissie in de aula van de Universiteit op

vrijdag 22 april 2016 om 14.15 uur

door

Sybren Huijink

geboren op 24 december 1988 te Doetinchem.

PROMOTIECOMMISSIE:

PROMOTORES:	prof. dr. Goos Kant
	prof. dr. Peter Borm
COPROMOTOR:	dr. ir. ing. René Peeters
OVERIGE LEDEN:	prof. em. dr. dr. h.c. Bernhard Fleischmann
	prof. dr. Joaquim dos Santos Gromicho
	prof. dr. Tom van Woensel
	prof. dr. Henk Zijm
	dr. Arantza Estévez Fernández
	dr. Marco Slikker

This research has been partially funded by the Dutch Institute for Advanced Logistics (DINALOG), within the context of the 4C4D: City Distribution project.

Preface

This preface tells the story of how I became a PhD-student and describes my time at Tilburg University. If this is not your cup of tea, I recommend the Introduction as a starting point. For the Dutch readers there is a summary at the end (Samenvatting).

At the age of six my parents already told me that I would become a doctor. At that time I 'learned', as all Dutch six-year-olds, to write italics. This style of writing has the disadvantage that it takes a while to write a sentence. Hence, like most physicians, I 'perfected' my style to a quick, but sadly almost ineligible handwriting. Since I already met this term for studying medicine, my parents thought that this profession suited me. Alas, as this dissertation suggests, it was not to be, but I have to admit that my parents were right in some respect as many members of this department are notorious for their handwriting.

Anyway, back to the start of my academic career. In my second year of studying at the university I became a student-assistant giving tutorials to students. My thanks to Marieke, Ruud, and Susan, for making me a better teacher. At the same time I discovered that I liked Game Theory and Combinatorial Optimisation. Before the start of the bachelor thesis season I went to Peter as I wanted to do a thesis in game theory. Peter, assisted by Ruud and Edwin, supervised me on the "The nucleon revisited" of which some results can be found in Chapter 8. Just before the master thesis season I visited Peter again. This time Peter and Herbert supervised me on sequencing games. Unfortunately, I suddenly had to teach a course and I utterly failed the assignment. The teaching, however, made me consider teaching at the university as a profession. Therefore I needed to do a PhD and to increase my chances I started a research master. However, I still had to do a master thesis. Surprisingly, Peter was willing to give me another shot, this time with John replacing Herbert. At the time I was finishing work on "Computational aspects of the per capita nucleolus", I was recruited by Goos and René to become their PhD-student for the Dinalog 4C4D project. The topic of this position at Tilburg University was to research cooperation between transportation companies. Immediately at the start, the project received its first setback. The goal was to start research on day one, but we made a wrong assumption, namely, I still had to finish the research master. Instead of switching to courses that left time for research, I stubbornly decided to stick with the challenging LNMB-PhD courses. This left me with just enough time and energy for teaching and interviews with the transportation companies. My thanks to the companies and their employees for their time and their valuable input.

During the first project of my PhD, I, together with Hans, John, and Peter, solved the conjecture of my master thesis. This resulted in the research master thesis "Computation of the per capita nucleolus in bankruptcy setting". The results of the two master theses can be found in Chapter 6 and Chapter 7. By this time, I had come to realize that I did not like the research for the project while I knew that I liked game theory. At some point I even wanted to quit and switch to game theory. A switch was not possible, which actually turned out well. Goos, René, and I, with a lot of help from Peter, found research which fitted both the project and my interests. I actually quite enjoyed thinking about methods to find solutions for the stylized planning of transportation companies and programming them. The results of the research of our project can be found in Chapters 2-5. I would like to thank my co-authors Goos, Hans, John, Peter, René, and Ruud, for their many corrections to avoid sloppiness and to make my work more readable (getting rid of the infamous 'Sybren-sentences'). Furthermore, they also slowed me down or pushed me ahead when needed. They also supported me in my private life, even though it provided setbacks in our research. They supported me when I got married and planned a long vacation when research-wise it was not the smartest move, and above all, granted me a leave for two months when it was needed due to unexpected circumstances. I would also like to thank my committee members Arantza, Bernhard, Henk, Joaquim, Marco, and Tom, for their serious efforts in improving my dissertation.

During my career I also met a lot of colleagues who indirectly influenced my dissertation. Let me begin with my roommates, first Aida, then Uwe, and finally Xingang. Thank you for the short intermezzos during work and making the office a nice place. Then there are the neighbors and the lunch-group members, Mario, Marleen, Marieke, Nick, Niels, Stefan, and many more. Thank you for the discussions during the lunches we had together. And finally let me thank the ones who answered all questions or redirected me to the right place: the secretaries Anja, Heidi, Korine, and Lenie.

Life is not only about work, and the support from my colleagues was magnified by the support I got in my personal life from my friends and family. First of all, I would like to express my gratitude to my parents Tom and Karin for their support and love, even though the path I took seemed to lead me away from their joke that I would become a physician. My brothers for all their support and the good times we had. They were also so kind to agree to be my 'meerkats' (also known as paranymphs). My good friends Art-Jan, Joost, Joost, and Oscar, for all the fun we had. Finally, I would like to express my gratitude to my lovely wife Susan, who I met at the first year of my study. She supported me through everything. Again, my gratitude to all.

Contents

Chapter 1

Introduction

1.1 Cooperation between transportation companies

This dissertation discusses cooperation between transportation companies. Increased restrictions and wishes imposed by customers and governments, together with increased competition, decreased the profit margins of transportation companies (Dahl and Derigs (2011); Stenger, Schneider, and Goeke (2013)). This has forced transportation companies to work more efficiently. One way to achieve this is by cooperating with other transportation companies.

In the first part (Chapters 2-5), we investigate pricing based cooperation structures, which are based on real-world cooperations. In this structure, companies typically outsource the delivery of certain orders to each other if the other company can execute this delivery more efficiently. Outsourcing is not the only option for cooperation. For example, a higher level of cooperation can be achieved when the companies have a joint planning and delivery of orders. Transportation companies can also cooperate specific on other aspects, for example, the purchase of trucks and equipments, truck fuel or maintenance contracts. In the second part of this dissertation (Chapters 6-8), we analyze how we can allocate, in theory, the additional gains when companies are involved in a higher level of cooperation. Or, in the most extreme case, participate in a central planning structure. This is not only applicable in the transportation industry, but also in other industries.

The reason why companies cooperate on deliveries is illustrated by Figure 1.1. The goal of the company in Figure 1.1 is to deliver all the orders, denoted by the small letters $a - h$, such that the costs are minimized. Each route should start and end at the depot, denoted by D.

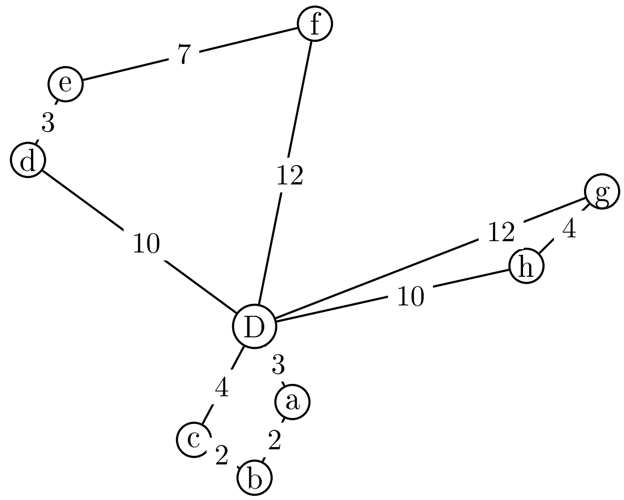

Figure 1.1: Example of the delivery of orders.

Suppose that the optimal way of delivering the orders consists of three routes ($DabcD$, $DdefD$ and $DghD$) which are shown in Figure 1.1. In this example there is a large difference between the length of the different routes (11, 32 and 26). The delivery of the orders that lie further away from the depot is more costly than delivery of orders that lie close. In order to decrease the costs, the company might consider cooperating with another company which has a depot closer to these far away orders. The cooperation does not only decrease the costs, but also has additional benefits. For example, there will be fewer truck movements in the region of the outsourced orders, which is especially important for urban areas given their focus on emission reduction. Various transport cooperations in the Netherlands, for example TransMission, Netwerk-Benelux, Teamtrans and Distri-XL, agree on a price for which members accept to deliver orders for each other. Based on this price, each individual company decides whether it delivers an order itself or outsources it. The decision is made by comparing the outsource costs with the routing costs. We call the problem of deciding which orders to outsource and which routes to drive the Vehicle Routing Problem with Order outsourcing. This problem was introduced by Chu (2005), while Hall and Racer (1995) were the first to decide which orders to outsource. The following example illustrates the decisions that have to be made in the Vehicle Routing Problem with Order outsourcing.

Example 1.1.1 Consider a company with the same eight orders, $a-h$, as in Figure 1.1. Figure 1.2 describes all the relevant distances between the orders. To deliver

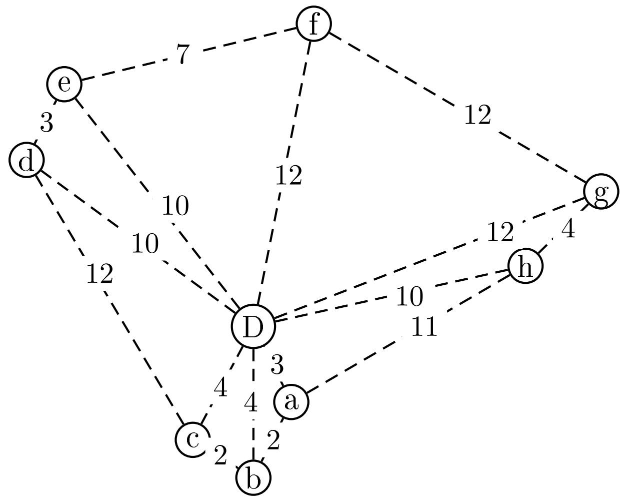

Figure 1.2: The distances between the orders.

the orders, the company has three trucks. Each truck has capacity 44, the fixed costs for using a truck are 12, and the variable costs per unit distance traveled are 1. The demand of the orders and the outsource costs of the full orders are given in Table 1.1. Note that the outsource costs of an order equals 4 plus the corresponding demand.

Order	a	b	c	d	e	f	g	h
Demand size	10	12	8	12	8	14	12	15
Outsource costs	14	16	12	16	12	18	16	19

Table 1.1: Characteristics of the orders.

The company has to decide which orders to outsource and on the routing of the remaining orders. For example, the company can decide to outsource all orders which costs 123 in outsource costs. Another example is to deliver all the orders as presented in Figure 1.1. Route $DabcD$ has a demand of $10+12+8=30$ and the corresponding costs are $12+(3+2+2+4)=23$. Similarly, $DdefD$ has a demand of $12+8+14=34$ and costs $12+(10+3+7+12)=44$. Finally, $DghD$ has a demand of $12+15=27$ and costs $12+(12+4+10)=38$. So, the total costs are $23+44+38=105$. However, for each route we do not only have the costs for delivering the orders, but also the costs when outsourcing the orders. The outsource costs are $14+16+12=42$, $16+12+18=46$ and $16+19=35$, for route $DabcD$, $DdefD$ and $DghD$, respectively. A comparison between the outsource costs and the routing costs shows that it is cheaper to outsource the orders in route $DghD$ than to deliver them. Note that it might be that it is cheaper to outsource a part of a route instead of the complete route. In fact it can be shown that the optimal solution is to outsource orders g and h, with costs 35, and to deliver the other orders, with costs $23 + 44 = 67$. Consequently, the total costs equal $67 + 35 = 102$. $\triangleleft$

In the second part of this dissertation, we analyze one of the most important reasons why cooperations fail. When companies fully cooperate on the actual joint planning and the joint delivery of orders, it is necessary to allocate the additional profits obtained by cooperating. The mistrust of the companies on this allocation has made many cooperations fail (Cruijssen et al. (2007)). Cooperative game theory addresses how to allocate the additional profits in a fair way. The following example illustrates the allocation issue with full cooperation on the delivery of orders between two companies.

Example 1.1.2 Consider two transportation companies, company X and Y. Company X has the same characteristics as in example 1.1.1, while company Y has 6 orders, $k - p$. For simplicity, the depot of company Y is at the same location as the depot of company X. Company Y has 2 trucks with the same capacity and cost characteristics as the trucks of the company X. The demand of all orders is given in Table 1.2.

Order	a	b	c	d	e	f	g	h	k	l	m	n	o	p
Demand size	10	12	8	12	8	14	12	15	14	10	8	10	16	17

Table 1.2: Characteristics of the orders.

In Figure 1.3, the two optimal individual solutions for the two companies (without cooperating and without the possibility of outsourcing) are provided.

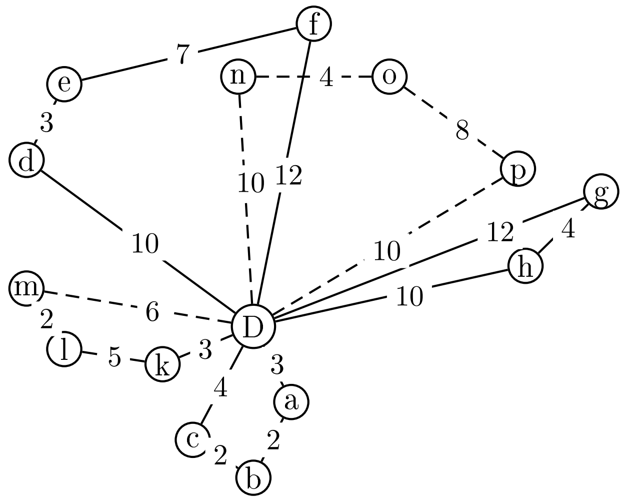

Figure 1.3: The separate delivery of orders.

The corresponding costs are 105 and 72 for the company X and Y, respectively, using a total of 5 routes. However, the companies only need 4 routes when they jointly plan and jointly deliver the orders. The four optimal routes when the companies fully cooperate are shown in Figure 1.4.

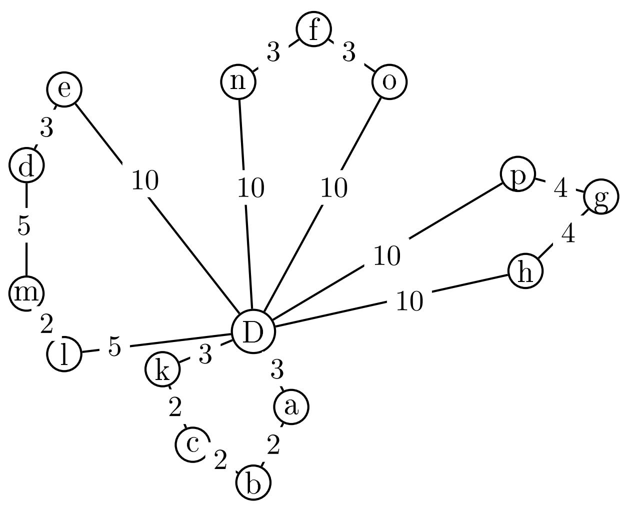

Figure 1.4: The joint delivery of orders.

The costs for delivering the orders when fully cooperating are 24 (route $DabckD$), 37 (route $DlmdeD$), 38 (route $DnfoD$), and 40 (route $DpghD$). So the total costs when the companies jointly plan and deliver are 139 while the total costs when the companies do not cooperate are $105 + 72 = 177$. The costs in this example are as follows:

S	$\{X\}$	$\{Y\}$	$\{X,Y\}$
$c(S)$	105	72	139

The question is how the costs, 139, can be fairly divided over the companies. One way to divide the costs is to use cooperative game theory. The game theoretic solution concept studied in Chapter 7, the per capita nucleolus (Grotte (1970)), assigns 86 to company X and 53 to company Y. Another option is to use the solution concept studied in Chapter 8, the proportionate nucleolus. The proportionate nucleolus assigns $82\frac{27}{59}$ and $56\frac{32}{59}$ to company X and Y, respectively. $\triangleleft$

1.2 Overview

In the first part of this dissertation (Chapters 2-5), we investigate (pricing based) cooperation structures between transportation companies with the focus on the delivery of orders. Chapter 2 discusses several cooperation structures for transportation companies in the context of opportunities and impediments for cooperating. An important opportunity is the reinforcement of the market position while an important impediment is the allocation of the joint gains (Dahl and Derigs (2011)). The two types of cooperation structures for transportation companies that are studied in the literature are central planning (Dai and Chen (2012)), where all the decisions are made by one company, and auction based (Berger and Bierwirth (2010)), where companies put undesired orders up for auction on which the others can place a bid. We analyze a new cooperation structure, which is the pricing based structure and is often used in practice. In the pricing based structure companies form a coalition and they outsource orders to each other within this coalition for given outsource costs per order. These outsource costs consist of the fee that the other company receives for the delivery plus the costs for the inter-depot transport. One of the most important decisions here is what the fee and the inter-depot costs of an order should be. However, the companies remain independent and decide themselves which orders they outsource. This implies that, in order to be able to determine the outsource costs that satisfy the preferences of the coalition, we need to know which orders the companies outsource given the outsource costs. We research two different methods that decide which orders to outsource, heuristics in Chapter 3 and estimations in Chapter 4. The companies do not know which orders will be outsourced to them when they have to make the decision which orders to outsource. Hence, we do not consider the orders that will be outsourced to the company when deciding which orders to outsource.

Chapter 3 discusses heuristics that decide which orders to outsource. Given the outsource costs, these heuristics try to find a set of orders to outsource together with routes for the remaining orders such that the total costs are minimized. We construct three large neighborhood search heuristics for the Vehicle Routing Problem with Order outsourcing. We find that the test instances used in the literature are not meeting the real-life characteristics and, therefore, we introduce new instances. The computational results show that our heuristics perform better than the existing heuristics in the literature.

Chapter 4 also discusses which orders to outsource. However, instead of heuristics, estimations are used to determine a set of orders to outsource. The costs for delivering an order are estimated. Based on these estimations it is decided whether or not an order is outsourced. We present a new type of estimation together with a new method to select which orders to outsource. Computational results show that our estimation method has better results than the existing estimations in the literature.

In Chapter 5, we focus on determining the outsource costs that minimize the total costs of the coalition. The outsource costs depend on certain factors, for example the demand of an order. The parameters of these factors are determined using Kriging (Forrester et al. (2008)). Kriging is a general method to find the best parameter configuration of a given problem. Using the results of several different parameter configurations, Kriging creates a model that approximates the outcomes of all parameter configurations. Iteratively, the most promising parameter configuration is calculated, using the heuristics in Chapter 3, and the approximation model is updated. We define a cooperation test instance for which we apply the pricing based structure. The computational results show, for this test instance, that the pricing based structure leads to a cost reduction of 14% compared to the case in which the companies do not cooperate.

In Chapter 2, we find that one of the most important impediments for cooperation structures, also in transportation, is how to allocate the additional gains. Chapter 6 starts with a summary of several allocation methods discussed in the literature on cooperation in logistics. Many of these methods use the cooperative transferable utility game model as a basis. The second part of this dissertation (Chapters 6-8) focusses on two allocation methods, the per capita nucleolus (Grotte (1970)) and the proportionate nucleolus. These two allocations are both variants of the nucleolus (Schmeidler (1969)). The nucleolus minimizes the maximal dissatisfaction of coalitions over all allocations, where the dissatisfaction of a coalition in a given allocation is expressed as the difference between the worth of the coalition and what is allocated to it. The per capita nucleolus minimizes the maximal dissatisfaction per player of a coalition. In other words, for the per capita nucleolus the dissatisfaction is shared over the players in the coalition while the nucleolus is interested in the dissatisfaction of the coalition as a whole. The proportionate nucleolus is a new concept (in the spirit of the nucleon (Faigle et al. (1998)) which maximizes the relative satisfaction over all allocations, where the relative satisfaction given a specific allocation is expressed as the ratio between the joint payoff that the coalition receives according to this allocation and the worth of the coalition. Chapter 7 characterizes the per capita nucleolus of bankruptcy games. Finally, Chapter 8 analyzes the proportionate nucleolus.

Chapter 2

Cooperation structures in logistics

2.1 Introduction

Cooperation structures between companies is a research field on its own (cf. Tjemkes et al. (2012)). We focus on cooperation structures in logistics. There exist several papers in the literature on this topic, both theoretical as well as case studies. In Section 2.2, we discuss the reasons why transportation companies cooperate (opportunities) as well as the obstacles (impediments). Many of these opportunities and impediments also hold for other sectors. Section 2.3 discusses the different cooperation structures in the literature and we briefly introduce a new structure, the pricing based structure. In Section 2.4, we discuss the pricing based structure more in depth.

2.2 Opportunities and impediments

Generally speaking, there are three reasons for companies to cooperate: reinforce the market position, reduce costs, and increase the service level. Reinforcing the market position (Krajewska and Kopfer (2006); Dahl and Derigs (2011); Wang and Kopfer (2011)) is the main reason and the other two reasons can be viewed as ways to achieve a reinforcement of the market position. From these two subreasons, the one which is quoted most is cost reduction (Cruijssen et al. (2007); Bloos and Kopfer (2009); Özener et al. (2011); Wang and Kopfer (2011)). Several case studies in the literature show that the possible savings lie between $5 - 25\%$ (Verdonck et al. (2013)). Note that these numbers highly depend on the characteristics of the companies and the cooperation structure. The final reason mentioned in the literature is service level improvement (Cruijssen et al. (2007); Bloos and Kopfer (2009)). The opportunities, the ones from the literature together with the ones we found during our interviews with transportation companies (see Appendix A), are presented in Table 2.1.

Opportunities	*Authors*
Reinforce market position	Krajewska and Kopfer (2006); Dahl and Derigs (2011); Appendix A
Cost reduction	Cruijssen et al. (2007); Bloos and Kopfer (2009); Özener et al. (2011); Wang and Kopfer (2011); Appendix A
Improved service	Cruijssen et al. (2007); Bloos and Kopfer (2009); Appendix A

Table 2.1: The opportunities of logistic cooperations.

Naturally, there are also impediments for the success of a cooperation structure. We divide the impediments into two types. The first type consists of the impediments concerning the selection of potential partners. The selection impediments cited the most are the unwillingness to disclose information (Berger and Bierwirth (2010); Özener et al. (2011); Wang and Kopfer (2014); Appendix A), doubts whether there actually exist any potential for savings (Cruijssen et al. (2006); Dahl and Derigs (2011); Appendix A), and the lack of trust in each other (Dahl and Derigs (2011); Özener et al. (2011); Appendix A). A new impediment is the lack of respect and vision. Some companies do not want that other companies also benefit from the cooperation and some companies rather lose money than transforming their business. A more complete list of the selection impediments can be found in Table 2.2. For an explanation of the other impediments we refer to their respective papers.

The second type of impediments concerns the operational impediments and the most cited one is how to share the benefits (Cruijssen et al. (2007); Lui et al. (2010); Dai and Chen (2012)). A new impediment is order discrimination. Orders should be delivered without differentiation between initial customers and those adopted from coalition partners, but some companies give their own customers a preferential treatment. A more complete list of the selection impediments can be found in Table 2.3. For an explanation of the other impediments we refer to their respective papers.

Selection impediments	*Authors*
Unwillingness to disclose information	Berger and Bierwirth (2010); Özener et al. (2011); Wang and Kopfer (2014); Appendix A
Low potential for savings	Cruijssen et al. (2006); Dahl and Derigs (2011); Appendix A
Bad geographical focus	Cruijssen et al. (2006); Appendix A
Lack of trust	Dahl and Derigs (2011); Özener et al. (2011); Appendix A
Low liquidity/ solvability	Cruijssen et al. (2006); Appendix A
Lack of respect and vision	Appendix A
Opportunistic behavior	Wang and Kopfer (2011); Appendix A
Loss of clients to partners	Cruijssen et al. (2007); Appendix A

Table 2.2: The partner selection impediments of logistic cooperations.

Operational impediments	*Authors*
Dividing the gains (Compensation scheme)	Krajewska and Kopfer (2006); Cruijssen et al. (2007); Lui et al. (2010); Dahl and Derigs (2011); Dai and Chen (2012); Appendix A
Determining savings	Dahl and Derigs (2011); Wang and Kopfer (2011); Appendix A
Communication	Dahl and Derigs (2011)
Increasing complexity	Frisk et al. (2010); Özener et al. (2011)
ICT	Dahl and Derigs (2011); Appendix A
Balancing of the workload	Cruijssen et al. (2007); Appendix A
Order discrimination	Appendix A

Table 2.3: The operational impediments of logistic cooperations.

2.3 Cooperation structures for transportation companies

Verdonck et al. (2013) state in a recent survey of cooperation structures in logistics that "Selecting an appropriate collaboration approach (cooperation structure) may be based on the type and amount of information organisations are willing to share with their partners, their experience with certain solution methods and so on." However, nowhere in their survey do they state how much information is shared in each cooperation structure or the amount of independence left for the companies. There are two main cooperation structures in the literature. One is central planning

(Krajewska et al. (2008); Lui et al. (2010); Dai and Chen (2012)) which is presented in Section 2.3.1. In this structure, all information is shared and all decisions are made by a central planner. In Section 2.3.2, we discuss the second main structure, the auction based structure. This cooperation structure uses auctions to determine which company will deliver which order (Schönberger (2005); Dai and Chen (2011)). Not all information is shared. However, more than the minimum is shared, where the minimum is to share the necessary information to the company that will deliver the order. In Section 2.3.3, we introduce a structure which is used in practice and only shares the minimum required information.

2.3.1 Central planning structure

In the central planning structure all companies operate as one. A planning company, which could be an independent company or a member of the cooperation, uses all the customers requests and all the resources, for example trucks, to generate an 'optimal' plan. This plan is then executed by all the companies. The central planning structure implies that the companies have to share all their data with the planning company and the companies do not have any decision freedom left with respect to which orders to outsource or accept. This is often not a problem since the companies either cooperate on something that is not there core business, examples of this are furniture producers who cooperate on the shipping of the furniture (Audy et al. (2011)), or timber industry on the transportation of logs (Frisk et al. (2010)), or the companies are profit centers (subsidiaries) of a larger company (Krajewska et al. (2008); Dai and Chen (2012)). The planning company can theoretically present the plan with the best gains. However, generating a good plan becomes more difficult if the size of the problem is larger. Clearly, the size of the cooperative plan is much larger than the individual problems of each company. Furthermore, the central planning company may not have the specialized knowledge of the local companies. This structure is not for companies who do not want to share their information or who want to make their own decisions. Furthermore, the additional gains still need to be divided fairly over the companies. This division is essential and a way to divide the gains is by using an allocation from cooperative game theory, which is the topic of Chapters 6-8.

2.3.2 Auction based structure

In the auction based structure each company first decides which orders it does not want to deliver with its own fleet. The undesired orders of all companies are put up for auction. Then, each company bids on (bundles) of orders. For example, Berger and Bierwirth (2010) auction off the undesired orders, one by one, using a second price sealed bid auction, which is also known as the Vickrey Auction (Vickrey (1961)). In the second price sealed bid auction, each company places a closed bid and the winner pays the second highest price. It is well known that the optimal strategy in a Vickrey Auction is to bid your true value for the auctioned item, which is a desired property. However, due to the complexity of the vehicle routing problem, a company does not know in general its true value for a request. Furthermore, "economic efficiency is enhanced if bidders are allowed to bid directly on *combinations* of different assets instead of bidding only on individual items" (De Vries and Vohra (2003)), which is especially true in vehicle routing problems. Auctions which incorporate bidding on combinations are combinatorial auctions and are used by many papers (Schönberger (2005); Krajewska and Kopfer (2006); Gujo et al. (2007); Schwind et al. (2009); Berger and Bierwirth (2010); Wang and Kopfer (2014)). However, combinatorial auctions give rise to the following problems. Similar as in the central planning, a planning company, which could be a computer, is needed that determines which company gets which orders. Furthermore, the bid, or the additional gains compared to the bids on the individual orders, has to be divided over the orders. In general, a combinatorial auction does not have the truth-telling incentive. The Generalized Vickrey Auction (Mackie-Mason and Varian (1994)) has the truth-telling property, but it requires a high computational effort (Berger and Bierwirth (2010)). This is due to the winner determination problem, also known as the combinatorial auction problem (De Vries and Vohra (2003)). The combinatorial auction problem determines which company wins which orders, but that is difficult to compute. In order to reduce the complexity of combinatorial auctions, some of the undesired requests are bundled by the companies or the auctioneer and auctioned as one (Schwind et al. (2009)). A slightly different approach is taken by Dai and Chen (2011), who propose a price-setting-based auction. In this auction, the auctioneer, which is the company who owns the undesired order, states a price for which it wants to outsource the order (or a set of orders) and the other companies determine whether or not they accept this price. The price is increased if no company is willing to deliver the order for this price and decreased if multiple companies are willing to deliver the order. The auction is stopped when there is exactly one company that is willing to deliver the order for the current price. Similar as in the central planning structure, the additional gains need to be divided over the companies when a combinatorial auction is used. This division is the topic of Chapters 6-8.

2.3.3 Pricing based structure

The pricing based structure has been in use for several decades by transportation companies in the Netherlands, examples are the alliances TransMission, Teamtrans, Netwerk Benelux and Distri-XL. The members of the alliance agree upon a pricing mechanism which specifies the costs for outsourcing an order to the other members. These outsource costs consist of the fee that the other company desires for the delivery of the order together with the costs for transporting the order between the depots. Together with the pricing mechanism, the companies agree upon a home-region for each company. Each company is obliged to deliver the outsourced orders that lie in its home-region. Naturally, these outsourced orders need to be transhipped between the depots. This inter-depot transport is planned by the alliance. Each member decides on its own, and solely based on its own requests and the outsource costs, which orders to outsource. After this decision, the outsourced orders are transshipped and each company plans the routing of its own orders that it did not outsource together with the orders that are outsourced to the company. The outsource costs are a critical factor for the gains. Outsource costs which are too high will result in not outsourcing orders which could be more efficiently delivered by the other company, while outsource costs which are too low will result in outsourcing orders which could be more efficiently delivered by the company. To the best of our knowledge there are no papers on the order outsourcing structure. However, there are two papers that work with proposals, but these papers assume that all the data is shared to a planning company. Dahl and Derigs (2011) and Özener et al. (2011) investigate a structure where the planning company generates proposals which the companies accept or reject. One of the main findings is that a wrong reward scheme results in the failure of the cooperation structure. We will discuss the pricing based structure in depth in Section 2.4.

2.3.4 Comparison between the structures

We distinguish between four important characteristics of the three different structures, namely, information sharing, decision freedom, computational complexity, and dependency on the decision who delivers which order. In the central planning structure all the information is shared to a planning company while in the pricing based structure only the information of the orders that are outsourced is shared. Moreover, in the pricing based structure the order information is only shared with the company that actually delivers the orders. The auctioning structure shares more information than the pricing based structure since all companies know the undesired orders. Furthermore, it might be that some undesired orders are not auctioned off. Due to these reasons, we set the amount of information shared to high, low, and medium, for the central planning, auction, and pricing based, respectively. The second characteristic is decision freedom. The companies have no freedom in the central planning structure as opposed to the auctioning structure where the companies can choose which orders to put up for auction and which orders to bid on. Note that there is no guarantee that the orders are auctioned off and there is no guarantee that the company wins the orders on which it bids. In the pricing based structure the companies have the freedom to decide which orders to outsource (and a guarantee that these orders are indeed outsourced) but they have the obligation to deliver the orders outsourced to them. Therefore, the decision freedom is low, high, and medium, for the central planning, auction, and pricing based structure, respectively. The daily planning complexity differs for all structures. In the pricing based structure companies need to determine, without any knowledge of the orders from other companies, which orders to outsource. After the transshipment, each company plans its total routing. The pricing mechanism is determined for a long period as well as the inter-depot transport which is adjusted when necessary after the outsourcing is determined. The details of the inter-depot transport and the assignment of the inter-depot costs to the orders are discussed in Chapter 5. The companies in the auction based structure need to decide which orders to outsource, on which to bid, the winner determination problem needs to be solved, and finally which routes to drive. In the central planning structure the planning company needs to determine all at once who delivers which orders, the inter-depot transportation, and the routes. The size of the problem in the central planning structure is much larger than the problem of a company in the outsourcing of orders, which, together with the dependence of the inter-depot transport planning and the decision which company delivers which orders, makes the central planning more difficult to solve than the outsourcing of orders. Similarly, the two additional problems, determining a bid on all the orders and the winner determination problem, makes that the computational complexity of the auction based structure is in between the other two. The final characteristic is dependency in the decision process. For the pricing based structure there is no dependency in the decisions which orders to outsource, since there is no information on the orders from the other companies. Furthermore, there are no daily decisions on which orders to deliver additionally. However, for both the auction based structure as well as the central planning structure the decision which orders to outsource and which to deliver are highly dependent on the orders of the other companies. Therefore, the decision dependency is high, high, and low, for the central planning, auction, and pricing based structure, respectively.

Cooperation	*information sharing*	*decision freedom*	*computation complexity*	*decision dependency*
Central planning	high	low	high	high
Auction based	medium	high	medium	high
Pricing based	low	medium	low	low

Table 2.4: Comparison of the different cooperation types.

Naturally, the central planning structure has the advantage that it (theoretically) can obtain the highest gain. However, depending on the instance and the heuristics used, it is possible that a heuristic for the other structure finds better solutions than a heuristic for the central planning (Wang and Kopfer (2014)). The auction based structure obtains 78% (Berger and Bierwirth (2010)) of the gains that the central planning structure would obtain, while the proposal method obtains 98% (Berger and Bierwirth (2010)). It is important to keep in mind that these results should be considered with some care. Due to the nature of the problems we are forced to use heuristics rather than exact methods to find solutions. Furthermore, the results depend on the (type) of instance, especially due to the optimization on sets of instances.

For the remainder of the first part of this dissertation (Section 2.4 - Chapter 5), we focus on the pricing based structure. The pricing based structure scores best on both information sharing and computation complexity, which are two important impediments.

2.4 Pricing based structure

Companies want to reduce their joint operational costs by efficiently delivering orders for each other. However, it is important for the companies to remain as independent as possible. The companies prefer that they choose themselves which orders to outsource. Furthermore, they are reluctant to share their daily operational information. Examples of cooperations that satisfy these constraints are the Dutch alliances which operate as follows. The members decide on an outsource pricing scheme for which companies can outsource orders to each other. This pricing scheme consists of the fee that the company that delivers this order receives together with the inter-depot costs of this order. Together with this scheme, the members agree upon a home-region for each company. Each company is obliged to deliver the outsourced orders that lie in its home-region. Finally, the members will plan centrally the inter-depot transportation. The goal of the members is to find the pricing scheme which minimizes the joint total costs, where the total costs of a company are its routing, inter-depot and outsource costs minus the fee that it receives. This is formally defined as follows. Let N be the set of companies, let $\mathbf{p}(.)$ be the pricing scheme, which denotes a function, and let $Costs_i(\mathbf{p}(.))$ be the total costs of company $i \in N$ given the pricing scheme $\mathbf{p}(.)$. Then, the objective of the alliance would be

$$\min_{\mathbf{p}(.)} \sum_{i \in N} Costs_i(\mathbf{p}(.)).$$

Note that the costs depend on which orders are outsourced by the companies, but what the companies outsource depends in turn on the pricing mechanism. Therefore, we start by discussing several methods for a company to determine what to outsource. For these methods we assume, for the time being, that the pricing scheme $p(.)$ is fixed and known. In other words, the companies know in advance the outsource costs of each order. In Chapter 5, we discuss how to find the pricing scheme $\mathbf{p}(.)$ such that the joint operational costs are minimized. However, we first discuss the daily operational decisions that each company has to make.

Daily planning for each company:
Given the outsource costs for each order, decide which orders to outsource (without knowledge of the orders of other companies).
(The outsourced orders are transshipped.)
Create a routing which delivers both the orders that the company did not outsource and the orders outsourced to the company.

When determining which orders to outsource, each company makes a trade-off between the costs of delivering an order with its own fleet and outsourcing it to the other member, responsible for the region where this order is located. The problem which orders to outsource and which orders to deliver can be modelled as the Vehicle Routing Problem with Order outsourcing (VRPO). Note that the orders of the other companies are not known when the company decides which orders to outsource, this includes the orders that will be outsourced to the company. Let $\{0, 1, ..., n\}$ be the set of orders where 0 stands for the depot. Each order $i \in \{0, ..., n\}$ has pre-determined outsource costs p_i, which is incurred when this order is outsourced, a demand q_i, and a location. The distance from the location of order i to the location of order j is given by d_{ij}. The company has m, for simplicity, identical trucks with capacity Q, fixed costs per day F for using the truck, and variable costs v per distance unit that the truck drives. Let $\Omega = \{1, ..., |\Omega|\}$ be the set of all feasible routes. Set $a_i(r)$ equal to one if route $r \in \Omega$ delivers order $i \in \{1, ..., n\}$ and zero otherwise and let $c(r)$ denote the total cost of route r. The decision variables are x_r where $x_r = 1$ implies that route r is driven. This gives the following set-packing model:

$$\text{minimize} \sum_{r\in\Omega} c(r)x_r + \sum_{i=1}^{n} p_i\Big(1 - \sum_{r\in\Omega} a_i(r)x_r\Big) \tag{2.1}$$

$$\text{subject to} \sum_{r\in\Omega} a_i(r)x_r \leq 1 \qquad \forall i \in \{1, ..., n\} \tag{2.2}$$

$$\sum_{r\in\Omega} x_r \leq m \tag{2.3}$$

$$x_r \in \{0, 1\} \qquad \forall\, r \in \Omega \tag{2.4}$$

The restriction that at most m trucks are driven, (2.3), and the integer restrictions, (2.4) are the same as in the classical vehicle routing problem (VRP) (see the VRP model at the end of this section, (2.8) and (2.9)). The first difference is in (2.2) (see (2.7)), which states that each order is delivered at most once instead of exactly once. The second difference is a consequence of the first and is in the objective function, (2.1) (see (2.6)). Where in the VRP each order has to be delivered, there is now the option not to deliver an order and incur the cost of outsourcing the order, in other words, a penalty. Hence, the objective function, (2.1), minimizes the total routing costs and the outsource costs.

The VRPO, which recently received much attention in the literature, is introduced as the Vehicle Routing Problem with Private Fleet and Common Carrier

(VRPPC) by Chu (2005), although the problem which orders to outsource, without routing the others, was already introduced by Hall and Racer (1995). Chu (2005) introduced the VRPO with a heterogenous fleet, but this is omitted in the model for clarity. Bolduc et al. (2008) showed that the VRPO can be rewritten as a Heterogenous VRP. Extensions of the VRPO are: the private fleet should serve a predetermined amount (or ratio) of the total demand (Tang and Wang (2006); Stenger, Schneider, and Goeke (2013)); that each private fleet vehicle can be used at most $t_{\max}$ time units (Stenger, Schneider, and Goeke (2013); Stenger, Vigo, Enz, and Schwind (2013)); dealing with multiple own depots (Stenger, Vigo, Enz, and Schwind (2013); Stenger, Schneider, and Goeke (2013)); non-linear costs and a capacity restriction for the common carrier (Stenger, Schneider, and Goeke (2013)); and hiring trucks and drivers (Wang et al. (2014)).

A mathematical equivalent problem of the VRPO is the Capacitated Profitable Tour Problem (CPTP) introduced by Archetti et al. (2009). In the CPTP, each order has a reward z_i which is collected when this order is visited by a private fleet vehicle, while in the VRPO an outsource cost p_i is incurred when an order is assigned to be outsourced. The CPTP has the same constraints as the VRPO (equation (2.2) up to (2.4)) but the objective function (equation (2.1)) becomes

$$\text{maximize} \sum_{i=1}^{n} z_i \sum_{r \in \Omega} a_i(r) x_r - \sum_{r \in \Omega} c(r) x_r. \tag{2.5}$$

The two problems are equivalent since taking $z_i = p_i$ will yield the same solution and the solution value can be recovered by the relation $VRPO = -CPTP + \sum_{i=1}^{n} z_i$, where $VRPO$ and $CPTP$ represent the optimal value of the respective problem. Closely related problems to the CPTP are the Team Orienteering Problem (Boussier et al. (2007)), which has as objective to maximize the rewards collected together with a restriction on the tour length. The single-truck variant of the CPTP without capacity constraint is the Traveling Salesman problem with Profits (Feillet et al. (2005)). Problems closely related to the VRPO are to outsource complete routes instead of orders (Moon et al. (2012)), and the Hot Rolling Scheduling Problems (Tang and Wang (2006); Tang et al. (2009)), which have several additional restrictions regarding the production of slabs. For a broader overview of these different problems we refer to Archetti et al. (2014). Example 2.4.1 illustrates the VRPO.

Example 2.4.1 Consider a company with four orders, $a - d$, and a depot D. The demand of the orders and the outsource costs of the full orders are given in Table 2.5.

Order	a	b	c	d
Demand size	11	11	19	20
Outsource costs	10	10	20	14

Table 2.5: Characteristics of the orders.

To deliver the orders, the company has two trucks. Each truck has capacity 44, fixed costs for using the truck are 12, and the variable costs per unit distance traveled are 1. Figure 2.1 describes all the distances between the orders.

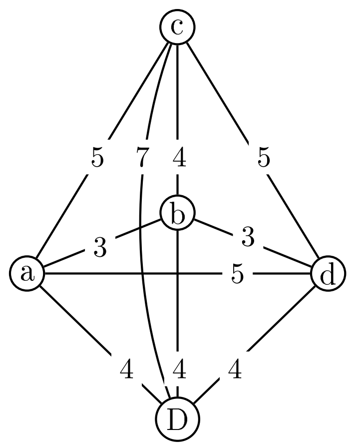

Figure 2.1: The distances between the orders.

This is a small example and all the 16 different solutions are enumerated in Table 2.6. The first column states which orders are delivered and the second column states the amount of routes together with the best routing. The distance is inferred from the routing, which is an optimization problem, and the outsource costs from the orders that are not delivered. The total costs is the sum of the routing costs and the outsource costs. In this example, there are many solutions that are worse than outsourcing all orders and only three solutions are better than delivering all orders. The best solution is to use one route which delivers the orders a, b, and c, and outsources the largest order, d. ◁

There are three different methods to determine which orders to outsource. The first method is to solve the VRPO to optimality by for example a branch-and-price algorithm (Archetti et al. (2013)). This branch-and-price algorithm will, if necessary, visit all possibilities and will provide a guaranteed optimal solution. However,

Orders delivered	Routes	Distance	Outsource costs	Total costs
∅	0	0	54	54
{a}	1 {DaD}	8	44	64
{b}	1 {DbD}	8	44	64
{c}	1 {DcD}	14	34	60
{d}	1 {DdD}	8	40	60
{a,b}	1 {DabD}	11	34	57
{a,c}	1 {DacD}	16	24	52
{a,d}	1 {DadD}	13	30	55
{b,c}	1 {DbcD}	15	24	51
{b,d}	1 {DbdD}	11	30	53
{c,d}	1 {DcdD}	16	20	48
{a,b,c}	1 {DacbD}	17	14	43
{a,b,d}	1 {DabdD}	14	20	46
{a,c,d}	2 {DacD}, {DdD}	24	10	58
{b,c,d}	2 {DbcD}, {DdD}	23	10	57
{a,b,c,d}	2 {DacbD}, {DdD}	25	0	49

Table 2.6: All the solutions.

it is time-consuming and is not suited for the size of instances in which we are interested. The second method is to use heuristics of which there are several available in the literature for the VRPO: Local Search heuristics (Chu (2005); Bolduc et al. (2007)); Tabu Searches (Côté and Potvin (2009); Potvin and Naud (2011)), Large Neighborhood Searches (Bolduc et al. (2008); Archetti et al. (2009); Stenger, Schneider, and Goeke (2013); Stenger, Vigo, Enz, and Schwind (2013)); and finally evolutionary (Kratica et al. (2012); Vidal et al. (2015)). Heuristics can not guarantee that the solution they find is optimal, but they are capable to find solutions for large instances and they are relatively less time consuming. We will discuss our heuristics in more detail in Chapter 3. The final method is to use estimations to make a trade-off between outsourcing and delivering an order (Hall and Racer (1995); Baghuis (2014)). The estimation method estimates the total costs that are assigned to an order, without actually constructing routes. Estimations take the least time to determine what to outsource but they are less accurate. In Chapter 4, we will discuss estimations in depth.

The first decision for each company is to decide which orders to outsource. The second decision is to decide on a routing for all its current orders, which are the orders that the company did not outsource together with the orders outsourced to the company. This can be modeled as a classical vehicle routing problem, which has the following set covering model:

$$\text{minimize} \sum_{r\in\Omega} c(r)x_r \tag{2.6}$$

$$\text{subject to} \sum_{r\in\Omega} a_i(r)x_r = 1 \qquad \forall i \in \{1, ..., n\} \tag{2.7}$$

$$\sum_{r\in\Omega} x_r \leq m \tag{2.8}$$

$$x_r \in \{0, 1\} \qquad \forall\, r \in \Omega \tag{2.9}$$

Since all orders need to be delivered, (2.7), the objective function is to minimize the total routing costs, (2.6). Note that the heuristics for the VRPO can also find solutions for the VRP by setting the outsource costs high for each order. For an overview of other solution methods for the VRP and its variants, we refer to Vidal et al. (2012). The pricing scheme automatically divides the gains over the companies since each company has outsource costs, receives fees, and has routing costs.

Chapter 3

Heuristics for the VRP with Order outsourcing

3.1 Introduction

This chapter, which is based on Huijink, Kant, and Peeters (2014), introduces several heuristics for the Vehicle Routing Problem with Order outsourcing (VRPO). In the VRPO each order receives an individual costs for which the order can be outsourced. The objective of the VRPO is to find the set of orders to outsource together with routes for the orders that are not outsourced such that the total costs are as small as possible. Due to the characteristics of the VRPO it is hard to find the optimal solution for reasonable sized instances. One method to generate solutions, although without guarantee that these are optimal, is to use heuristics. In Section 3.2, we discuss the main ideas behind the heuristics proposed for the VRPO. For a broader overview of heuristics for the classical vehicle routing problem and several of its extensions we refer to Vidal et al. (2012).

Our ultimate goal is to find the best pricing scheme (see Section 2.4), which implies that we need a heuristic which finds good solutions irrespective of the characteristics of the instance. It should find good solutions whatever the outsource costs are, and without the need for optimizing the heuristic on specific characteristics. The best heuristic known by us in the literature for the VRPO with a homogenous fleet, a genetic search heuristic, will not perform well when many orders are outsourced (Vidal et al. (2015)). Therefore, we have decided not to use this heuristic. Furthermore, the large neighborhood search heuristics for the VRPO have only one neighborhood structure and they have worse results than the tabu search heuristics. Preliminary testing showed that a simulated annealing approach (see Vidal et al.

(2012)) as well as a tabu search heuristic did not deliver promising results. Since we need a heuristic which performs well regardless of the characteristics, we have chosen for large neighborhood search heuristics that have several different neighborhood structures and use tabu search as a local improvement heuristic. We introduce several neighborhood structures in Section 3.3. In Section 3.4, we show that the instances commonly used in the literature have as characteristic that orders are unnecessarily outsourced. In other words, the prices are high and the orders are only outsourced since there is not enough capacity to deliver them. To amend this, we propose modifications of the original outsource costs. We show in Section 3.5 that our large neighborhood heuristics are currently the best heuristics for the VRPO, if tested on the original instances. Furthermore, we show that the different neighborhoods are important when there are many different pricing schemes. The conclusions and recommendations are presented in Section 3.6.

3.2 Heuristics: an overview

The first heuristics developed for the VRPO are improvement heuristics (Chu (2005); Bolduc et al. (2007)). Improvement heuristics consist of two phases, a construction phase which creates a solution, and an improvement phase, which improves the solution created in the construction phase. The basis of an improvement heuristic is presented in Algorithm 3.2.1.

Algorithm 3.2.1 *Improvement heuristics*

Setup phase

Generate an initial solution

Improvement phase

Repeat the following until there is no improvement available

Search the neighborhood for the best improvement and implement it

How to generate an initial solution and what the neighborhood should be are important parts of the improvement heuristic. Many heuristics for the VRPO have the following structure: First, it selects which orders to deliver (hence, not outsourced) and then uses a construction method for composing routes for these selected orders. The selection is based on the outsource costs and the demand of the orders. The orders that have the lowest ratio of its outsource costs divided by its demand will be outsourced until the remaining orders will not have a total demand more than the available capacity. This implies that there will be no orders outsourced when there is enough capacity available for all the orders. After the selection, the heuristics use the well-known Clarke and Wright savings algorithm, which is a construction heuristic for the classical vehicle routing problem. The Clarke and Wright savings algorithm starts by assigning each order to the routes such that each route has exactly one order. Then, the savings algorithm merges greedily, where greedy implies that the best merges is chosen at each step, until there is no feasible merger left.

An exemplary neighborhood for improvement heuristics is the neighborhood consisting of all the 1−*shift* moves and 1−*swap* moves. The 1−shift move shifts one order from one route to another, in other words one order is removed from a route and inserted into another route. The 1−swap move swaps two orders from different routes. The 1−shift and 1−swap moves are illustrated in Figures 3.1 and 3.2. Finally, the best move is the move which will result in the least additional costs. A move is an improvement if the additional costs are negative. The improvement heuristic will stop searching when the best move is not an improvement. In other words, it ends in a local optimum. Often an instance has multiple local optima and it could be that another optimum has less costs than the one where the heuristic ends.

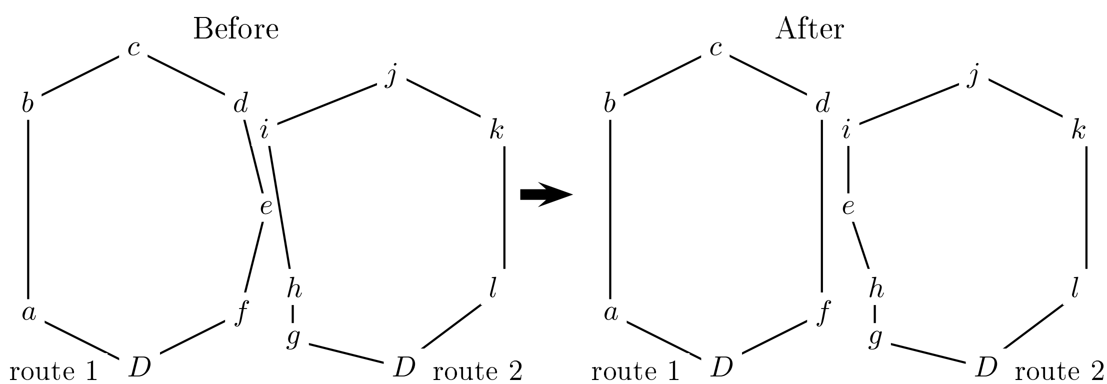

Figure 3.1: Example of the 1−shift move.

The tabu search heuristic is an extension of the improvement heuristic and it is able to escape local optima. As opposed to the improvement heuristic, the tabu search allows deteriorations of the solution and to avoid immediate returning to the previous solution it makes some moves *tabu*, which means that it temporarily forbids some moves. There are two tabu search heuristics for the VRPO (Côté and Potvin

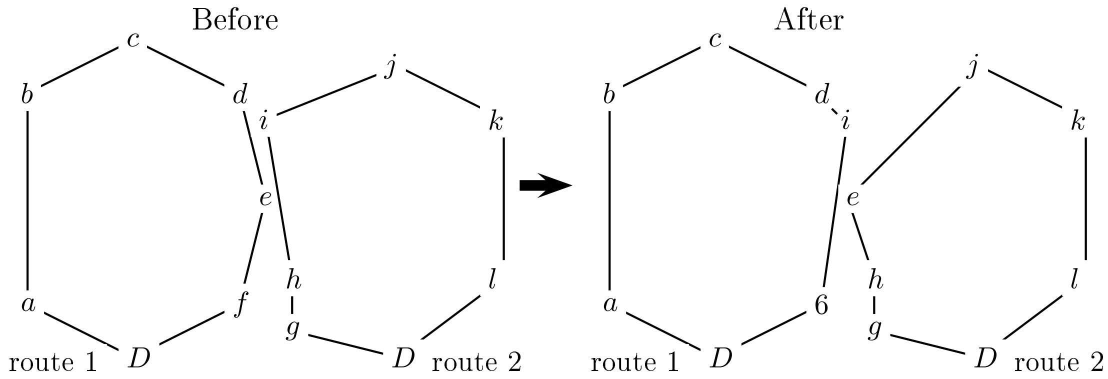

Figure 3.2: Example of the 1−swap move.

(2009); Potvin and Naud (2011)). The basis of a tabu search is shown in Algorithm 3.2.2.

Algorithm 3.2.2 *Tabu search heuristics*

Setup phase

Generate an initial solution

Improvement phase

Repeat the following until a stopping criterion is met

1: Search the neighborhood for the best move that is not tabu

2: Implement it and update the tabu-list

Similar as in the improvement heuristic there are decisions on which initial solution and which neighborhood to use. However, there are more decisions to be made in a tabu search heuristic. These decisions are: Which moves are forbidden after a move is implemented and for how long? When should the heuristic stop and finally, what to do if a move that would improve the solution to be better than the currently best solution is tabu?

The tabu search heuristic escapes the local optimum using many small steps. Another way to escape a local optimum is to jump to another solution. This is the idea behind the large neighborhood search heuristic, of which there are several for the VRPO (Bolduc et al. (2008); Archetti et al. (2009); Stenger, Schneider, and Goeke (2013); Stenger, Vigo, Enz, and Schwind (2013)). The large neighborhood search uses one of the previous two heuristics to improve the solution after the jump. By using ruin-and-repair moves instead of many small steps, the large neighborhood search heuristic is able to visit more different solutions. Algorithm 3.2.3 shows the basis of the large neighborhood search heuristic.

Algorithm 3.2.3 *Large neighborhood search heuristics*

Setup phase

Generate an initial solution

Improve the solution (using the improvement phase of the improvement or tabu search heuristic)

Improvement phase

Repeat the following until a stopping criterion is met

1: Ruin (and repair) the solution

2: Improve the solution

3: Decide to accept the new solution or revert to the solution before the ruin step

Additional decisions, compared to the previous heuristics, are how to ruin and repair the solution, and when to accept the solution. The goal is to visit new solutions, which implies that the solution after the ruin-and-repair move should not be too similar to the previous solution. On the other hand, moving too far from the previous solution can destroy parts of the solution which were good. We will discuss large neighborhood search heuristics in depth in Section 3.3.

The final type of heuristics that are used for the VRPO are evolutionary algorithms, more specifically, genetic heuristics (Kratica et al. (2012); Vidal et al. (2015)). The largest differences are that the genetic heuristic starts with multiple solutions and uses these solutions to breed new ones instead of working with one solution. The basis of the genetic heuristic is presented in Algorithm 3.2.4.

Algorithm 3.2.4 *Genetic search heuristics*

Setup phase

Generate initial solutions

Improve the solutions (using the improvement phase of the improvement or tabu search heuristic)

Improvement phase

Repeat the following until a stopping criterion is met

1: Select a set of solutions and discard the others

2: Generate new solutions from this set of solutions
3: Improve the new solutions and add (part of them) to the solutions

As before, one of the decisions is which initial solutions to use. Another decision is how to select the set of solutions, the so called survivors, which criterion should be used? From these survivors, new solutions are generated, which are called the offspring. Again, the goal is to create solutions which take the good parts from several older solutions.

3.3 New large neighborhood search heuristics

There exist two large neighborhood search heuristics for the VRPO. However, each of these heuristics only use one type of ruin-and-repair move. We present in this section three new large neighborhood search heuristics. Two of these new heuristics find better solutions than the two large neighborhood searches from the literature. Instead of using only one type of *ruin-and-repair* move (neighborhood), we use many different types of moves. The ideas of the new heuristics and the moves are presented in this section, while the details can be found in Appendix B.1. For each move we discuss whether it can also be used for other vehicle routing problems. For the moves we make a distinction between *driven routes*, which are routes that have less delivery costs than outsource costs of the orders in the route, and *candidate routes*, which are routes that have larger delivery costs than outsource costs of the orders in the route. In the same vein, we make a distinction between *delivered orders*, which are the orders that are in the driven routes, and *outsourced orders*, which are the orders that are in candidate routes or are not in any route.

3.3.1 Initial solution

Many heuristics for the VRPO use a method to select which orders to deliver and then use the classical savings or insertion algorithm. We use the *profitable insertion* method, which is the following adaptation of the insertion method: The classical insertion method iteratively inserts the undelivered order with the least insertion costs, where the insertion costs are the additional costs for delivering the order. However, in the VRPO it could be that this order should be outsourced. Hence, the adaptation uses the *profit* of an order, which is the outsource costs minus the insertion costs divided by the demand of the order. In the classical least insertion, the first order of a route is the order that lies the furthest from the depot. Again, it could be that this order should be outsourced. Therefore, we take the order with the highest profit as the first order.

3.3.2 Ruin-and-repair moves from the literature

The first type of ruin-and-repair move is the *remove-and-insert* move from Archetti et al. (2007). The move starts with randomly selecting a number of delivered orders and removes them from the driven routes. Then, the remaining outsourced orders, but not the ones which were just removed, are randomly selected until the total outsource costs of these selected outsourced orders is equal to the outsource costs of the removed delivered orders. The selected outsourced orders are inserted into the driven routes such that the *infeasibility*, which is the square of the amount of demand more than the available capacity of the route, and the insertion distance is minimized. The usage of orders that are outsourced in this move imply that it can only be used in vehicle routing problems where not all orders are delivered, for example the VRPO or TOP. Figure 3.3 illustrates the remove-and-insert move, where the dots represent the orders that are delivered, the squares the orders that are outsourced, the triangles the delivered orders that are removed, and the plus signs represent the outsourced orders that are inserted.

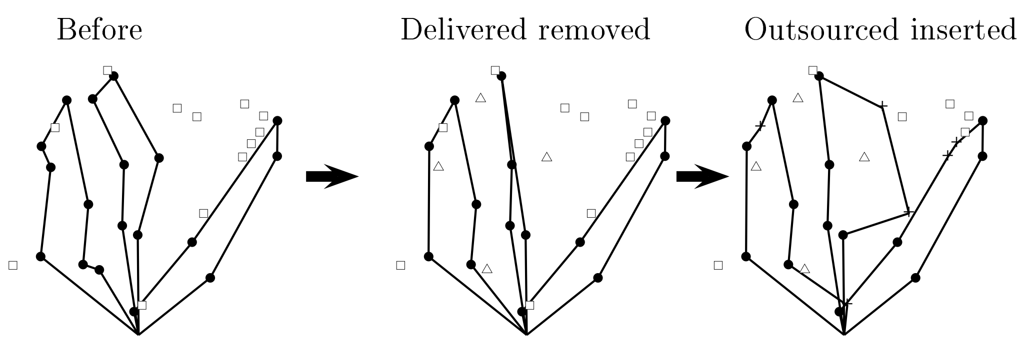

Figure 3.3: Example of the remove-and-insert move.

The idea behind the *cyclic* moves of Stenger, Schneider, and Goeke (2013) is to move parts of a route from one route to another. The cyclic move starts by selecting a route, which can be for example random or based on the distance to other routes. From this chosen route a part, which may consist of more than one order, is selected to be moved. This selection can again be for example random or based on the distance from the location of the order to the location of orders in other routes. These selected orders are moved to the other route which has the least insertion costs of these orders. Then, from this other route a part is selected to be moved. The cyclic move can be applied to all vehicle routing problems. However, in case of the VRPO, we use a cyclic move that moves orders in between driven routes and use an additional cyclic move that moves orders between driven and candidate routes. This cyclic move is illustrated in Figure 3.4, where we select the most left route and from this route we select the top part to be inserted into another route, which is in this case the middle route. From the middle route we select a part of the route and this is inserted into the left route. This example is only with two routes, which are both driven, but it can also be used with more routes or with a candidate route and a driven route.

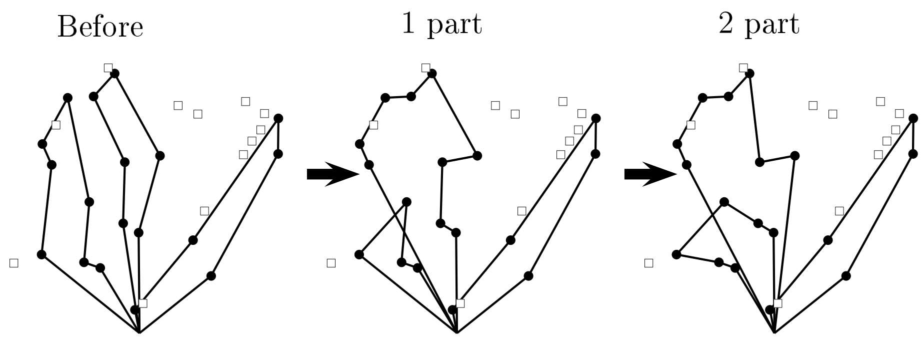

Figure 3.4: Example of the cyclic move.

3.3.3 New ruin-and-repair moves

The first of our new moves that we discuss is the *create* move which creates a new route. However, it is often not profitable to create a new route using only the orders that are outsourced. The other routes should make space for the new route. We start the creation of a new route by selecting an outsourced order and put it into an empty route. Next, we apply the repair procedure which we call *shifting*. The shifting procedure is very similar to the profitable insertion method (see Section 3.3.1), but has a few key differences. The routes which were changed in the move are called *receiving* routes and the receiving route in the first step of the create move is the route which was just created. Instead of selecting the outsourced order with the highest profit, we select the order so far not belonging to the receiving route with the highest profit. In other words, orders that belong to driven routes are also allowed to be inserted into the receiving route. However, we only allow a certain percentage of orders to be taken from each driven route. This will typically result in filled receiving routes but the other routes which were driven before we started the shifting procedure will be partially filled. To adjust this, we repeat the process a few times but now the driven routes that have a load factor below a certain percentage are the receiving routes. Furthermore, to avoid circular behavior we do not allow the insertions from previous receiving routes. In principle, the create move can be used for all vehicle routing problems. However, the goal of many VRP's is to minimize the number of routes, which implies that the create move is likely to be only useful for the TOP and VRPO. The shifting procedure can be used for all VRP's, even though our procedure is primed to use outsourced orders. Figure 3.5 illustrates the shifting move, where the dashed route is the first receiving route (and in this case the route created with the create move).

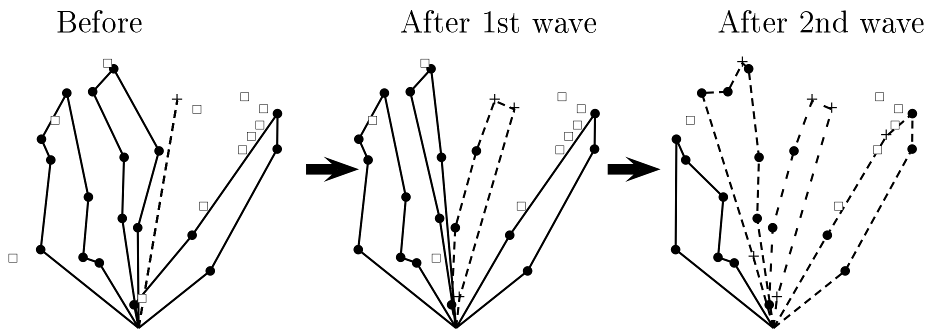

Figure 3.5: Example of the shifting procedure.

The *destroy* move selects a route and all the orders in the route are outsourced. The *split* move splits a route. It selects a route and moves a part of the route to an empty route. After the split, the shifting procedure is applied where the chosen route and the route that received a part of the route are the receiving routes. The destroy move can be used for all VRP's, although it will not be helpful if the solution before the destroy move already uses the minimal number of routes. The split move has the same issues as the create move. These moves are illustrated, without the shifting procedure, in Figure 3.6.

The *bomb* move randomly selects an order, which could be delivered or outsourced, to be the *center* of the bomb. The delivered order which has its location closest to the center is removed from the route and outsourced until a number of driven routes and a number of delivered orders are removed. After this the shifting procedure is used where all the routes that were modified are receiving routes.

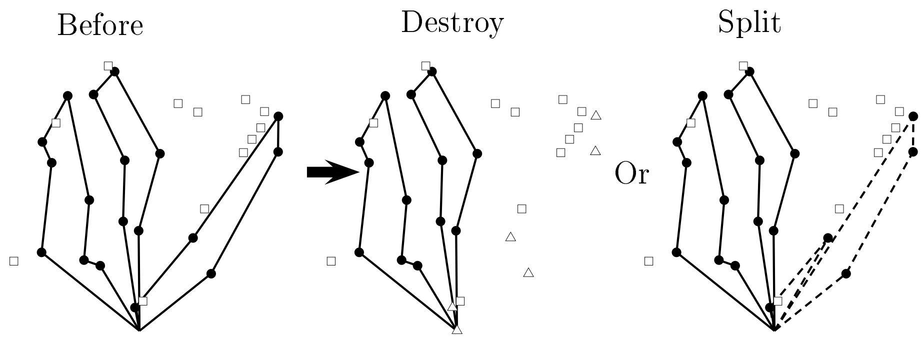

Figure 3.6: Example of the destroy and the split move.

The *reseeding* move selects several routes and removes parts of these routes. Again, the shifting procedure is used after the reseeding to repair the solution where the routes that were selected are the receiving routes. Both the bomb and reseed move can be used in any VRP. However, the repair procedure should be such that it does not return to the previous solution. The bomb and reseeding moves are illustrated, without the shifting procedure, in Figure 3.7, where the arrow points at the center of the bomb.

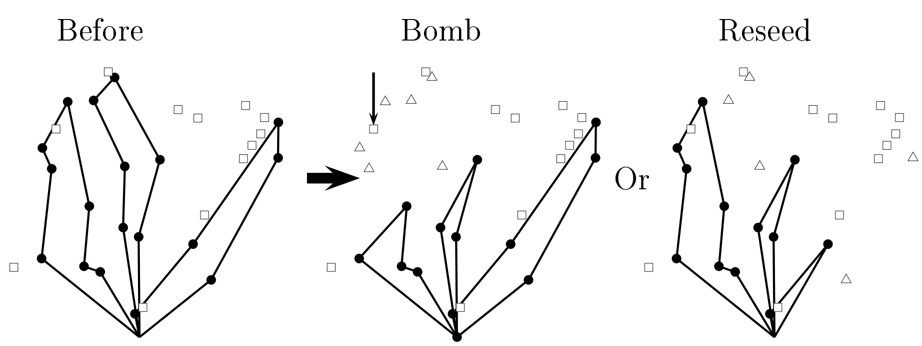

Figure 3.7: Example of the bomb and the reseeding move.

The final move is to apply the improvement method again.

3.3.4 Acceptance criterion

We not only accept a solution if it is better than the solution before the ruin-and-repair move, but if a certain number of moves did not find an improvement of the solution we also accept, with a certain probability, a deterioration. The reason behind this is that the ruin-and-repair moves were unable to improve the solution for a certain number of moves, in other words, the solution appears to be a local optimum. Hence, moving to another solution, which is in this case the deterioration, and applying the ruin-and-repair moves on this deterioration could lead to an improvement of the deterioration which is better than the solution before the deterioration.

3.3.5 Three heuristics

Using the different neighborhood moves discussed above, we create three different heuristics. In each of these heuristics we have an objective function which consists of three levels. The first level minimizes the total costs, which includes a penalty for the infeasibility in the driven routes, where the infeasibility of a route is given by the square of the capacity overload. The second level minimizes the total distance driven and the total infeasibility. Finally, the third level tries to make the candidate routes as profitable as possible. The first heuristic is a very simple large neighborhood search. One could even argue that it is not a real large neighborhood search but an enhanced tabu search. This heuristic, called the reseeding-tabu search (R-TS), is a large neighborhood search heuristic where we use a regular tabu search heuristic followed by two times the reseeding move and a tabu search. The R-TS is used as a baseline to compare heuristics. The second and third heuristic are large neighborhood search heuristics, the LNS-Fast and LNS-Slow, and both use all the ruin-and-repair moves discussed in Section 3.3. The main difference between the LNS-Fast and the LNS-Slow is that the LNS-Fast executes a move once and that the LNS-Slow executes the move five times.

Most heuristics are heavily influenced by the parameters. For example more iterations will increase the running time, but will probably also improve the final solution. Another example of parameters is how many orders to select in our bomb move since more orders will result in a more diversified solution. Or a long tabu duration moves the tabu search away from the current solution. Furthermore, the parameters for adjusting the infeasibility will guide the tabu search since a low infeasibility penalty will imply that the tabu search will visit infeasible solutions. In the literature, these parameters are often optimized over a training set of instances. However, there is no guarantee that the training set is representative for all instances. Furthermore, it is often impossible to analyze all possible different values of the parameters and hence a surrogate model, just like the one in Chapter 5, is used in the literature to optimize the parameters. Finally, we need a robust heuristic which can handle different instances. Therefore, we have opted for a different approach.

The parameters are drawn randomly from reasonable bounds. For example, the tabu search of Côté and Potvin (2009) has as tabu duration a random integer in $[7, 14]$ and the two update parameters for infeasibility are 1.25. Our tabu duration is a random integer in $[t_{lower}, t_{upper}]$, where t_{lower} is a random integer in $[4, 9]$ and t_{upper} is a random integer in $[12, 27]$, and both our update parameters are randomly drawn from $[0.5, 2]$. By doing so, our local search will be different each time.

Below we present an overview of our three heuristics, where Algorithm 3.3.2 describes two heuristics. The parameter settings can be found in Appendix B.1.

Algorithm 3.3.1 *Reseeding-tabu search*

Setup phase

Generate an initial solution

Improvement phase

Repeat the following until a stopping criterion is met

1: Search the neighborhood for the best move that is not tabu

2: Implement it and update the tabu-list

Reseeding phase

Repeat the following twice

3: Apply the reseeding move

4: Apply the improvement phase

5: Keep the new solution if it is better than the current best solution

Algorithm 3.3.2 *Large neighborhood search-Fast (and -Slow)*

Setup phase

Generate an initial solution

Improvement phase

Repeat the following until a stopping criterion is met

1: Search the neighborhood for the best move that is not tabu

2: Implement it and update the tabu-list

Ruin-and-repair phase

Repeat the following until a stopping criterion is met

3: Choose a ruin-and-repair move and execute it (5 independent times)

4: Apply the improvement phase (independently on all 5 solutions)

5: Decide to accept the new solution (best of the 5) or revert to the solution before the ruin method

3.4 Test instances

The test instances of Bolduc et al. (2008) are modifications of the test instances of Christofides et al. (1979) (CE) and Golden et al. (1998) (G). In the 14 CE-instances, the number of orders ranges from 50 to 199, the number of routes from 4 to 13, and the number of orders divided by routes ($\bar{n}$) from 8.3 to 20.0. For the 20 G-instances, the number of orders ranges from 200 to 480, the number of routes from 4 to 31, and the number of orders divided by routes from 12.0 to 60.0. Note that in the G-instances the first 12 instances have an $\bar{n}$ between 23 and 60, while the last 8 have a $\bar{n}$ between 12 and 17. The coordinates, demand and capacity are kept, but all the time restrictions are dropped. In other words, the route-duration (and hence length), time-windows and fixed time per stop, are dropped. Additionally, the number of available routes is set to $\lceil 0.8 \sum_{i=1}^{n} q_i \rceil$, where q_i is the demand of order i, and each route has a variable cost of 1 per unit distance. This implies that slightly more than 80% of the demand can be delivered by the own fleet. Furthermore, Bolduc et al. (2008) set the fixed cost of a route F to the average route length within the best known solution of the original instance with timing restrictions, rounded to the nearest integer which is divisible by 20. Finally, the outsource costs of each order is set as follows: Let $\bar{n}$ be as before and let $q_{\min}, q_{\max}$ the minimum and maximum demand of the orders, respectively. Set $\eta = \frac{q_{\max} - q_{\min}}{3}$ and

$$\mu_i = \begin{cases} 1 & \text{if } q_i \in [q_{\min}, q_{\min} + \eta), \\ 1.5 & \text{if } q_i \in [q_{\min} + \eta, q_{\min} + 2\eta), \\ 2 & \text{if } q_i \in [q_{\min} + 2\eta, q_{\max}]. \end{cases}$$

The outsource costs, or common carrier costs, are given by $p_i = round(\frac{F}{\bar{n}} + \mu_i \cdot d_{0i})$. In other words, the outsource costs consists of the average fixed costs incurred per order and the distance costs between an order and the depot times a factor which is based on the size of the order. An overview of these test-instances is presented in Bolduc et al. (2008). This pricing results in outsource costs that are higher than the costs when delivering the demand using the own fleet. This is reflected by the fact that in all except G-13, G-14, and G-16, the best known solution uses all the available routes. Moreover, in the solutions we found that the routes deliver on average more than 83% of the total demand. There are two variants of each instance, one with a homogenous fleet and one with a heterogenous fleet.

To amend the issue that the outsourcing is only done out of necessity, we modified the test instances by halving the outsource costs. This set of instances, called the Half-Original instances, are copies of the original instances of Bolduc et al. (2008), but now the outsource costs are halved, i.e., $p_i^{half} = \frac{p_i}{2}$. Furthermore, the instances with the original outsource costs, and by inheritance the Half-Original costs, have as additional characteristic that delivering orders with a small demand is preferred over larger ones. In our solutions we found that on average more than 90% of the orders are delivered, even while the total demand delivered is less than 84%. This means that an average order size of an outsourced order is 1.6 times the size of an average order. The difference is even larger for the Half-Orig costs, namely 75% of the orders are delivered while the demand delivered is only 65%. That orders with a larger demand are delivered, instead of the more intuitive outsourced, is a direct consequence of the involved pricing scheme and is illustrated in the following example.

Example 3.4.1 Consider the same company as in Example 2.4.1. For convenience we repeat the important parts. The company has four orders $a - d$, and a depot D. The demand of the orders and the outsource costs of the full orders are given in Table 3.1.

Order	a	b	c	d
Demand size	11	11	19	20
Outsource costs	10	10	20	14

Table 3.1: Characteristics of the orders.

To deliver the orders, the company has two trucks. Each truck has capacity 44, the fixed costs for using a truck are 12, and the variable costs per unit distance traveled are 1. Figure 3.8 describes all the distances between the orders.

The total distance when delivering all orders is 25, as is illustrated in Table 3.2. This gives, according to the settings in the original pricing, that the (rounded) fixed costs per route are 12 and the outsource costs $p_i = 6 + \mu_i \cdot d_{0i}$. This gives the following three best solutions (see Table 2.6 for a full enumeration) together with the solutions when everything is either outsourced or driven.

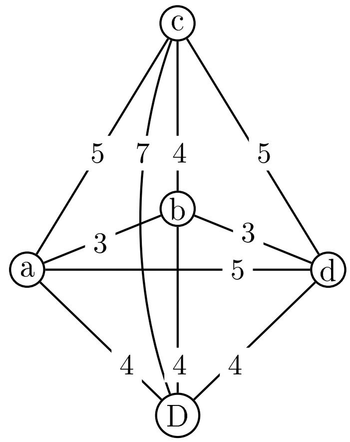

Figure 3.8: The distances between the orders.

Orders Driven	Routes	Distance	OutsourceCosts	Total Costs
∅	0	0	54	54
{c,d}	1	16	20	48
{a,b,c}	1	17	14	43
{a,b,d}	1	14	20	48
{a,b,c,d}	2	25	0	49

Table 3.2: The basic solutions and the three best.

This implies that delivering orders $\{a, b, c\}$ is the best one can do, in other words, choosing two orders with a small demand (a and b) instead of one with a large demand (d) even while the distance to deliver both a and b is larger. This is due to the fixed cost $\frac{F}{\bar{n}} = 6$ in the formula of the pricing together with that the other part of the pricing, in other words, $\mu_i d_{0i}$, cannot be more than twice as large for orders that have the same distance to the depot. In this case, order d has also the lowest ratio of outsourcing costs divided by demand. However, the following modification presented in Figure 3.9, where the location of order d is shifted further away, shows that there are also simple examples where the outsourced order has the highest ratio of outsource costs divided by demand.

The total distance when delivering all orders is now 33 see Table 3.4. There are two different solutions, one which has routes DdD and $DacbD$, and one which has routes $DcdD$ and $DabD$. Since we apply the original pricing (which uses the distance costs for delivering all orders), we have that the fixed costs per route are 16 and the adjusted outsource costs are presented in Table 3.3.

The numbers in the three solutions also change and are given below.

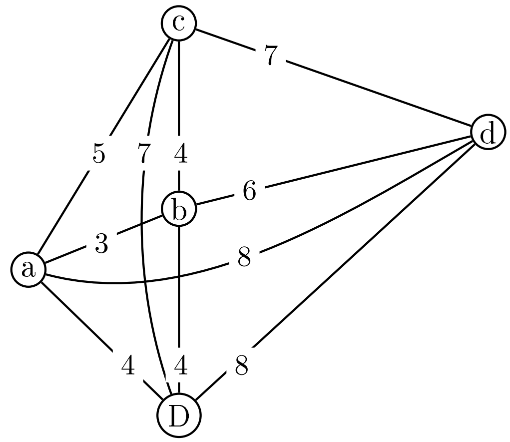

Figure 3.9: The orders and distances.

Order	a	b	c	d
Demand	11	11	19	20
Outsource	12	12	22	24

Table 3.3: Characteristics of the orders.

Orders Driven	Routes	Distance	OutsourceCosts	Total Costs
∅	0	0	70	70
{c,d}	1	22	24	62
{a,b,c}	1	17	24	57
{a,b,d}	1	21	22	59
{a,b,c,d}	2	33	0	65

Table 3.4: The basic solutions and the three best of the modified example.

Note that the optimal solution is again to outsource order d. ◁

Example 3.4.2 shows that the outsourcing in the Original costs is done out of necessity. Furthermore, it also shows why orders with a larger demand are outsourced, even though these larger orders have a much higher outsource costs than orders with a smaller demand.

Example 3.4.2 The CE-04 instance has 150 orders and 9 available routes. The costs of outsourcing an order depends on the fixed costs of a route and the distance of the location of an order to the depot. The (jigsaw) surface in Figure 3.10 is the estimated profit, which is the outsource costs minus the estimated delivery costs (see Chapter 4), of the orders. This figure shows that within each level of μ it is more profitable to deliver orders with a small demand. Furthermore, there is not enough capacity to deliver all the orders and two orders with a small demand will give a higher profit than one order with a large demand. ◁

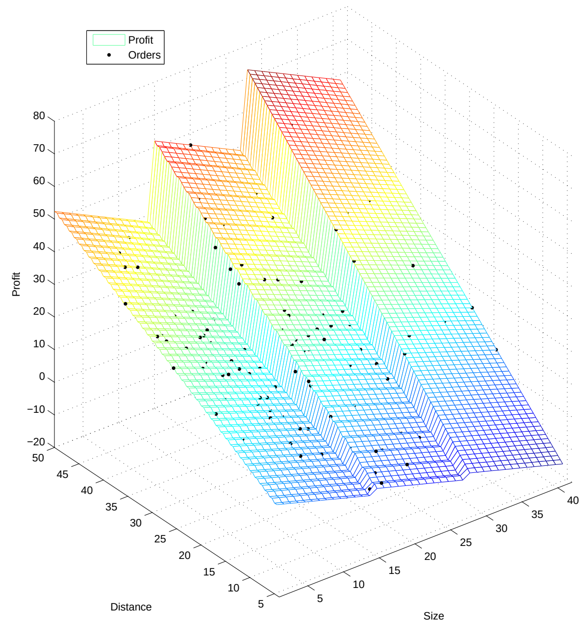

Figure 3.10: The estimated profit of delivering orders

To amend the issue that orders with a large demand are more likely to be outsourced and to avoid that the outsource costs are dependent on the distance of the location to the depot, we create a third set of costs. This set of instances, called the New instances, again copies the original instances, but now with the outsource costs as a fraction of the fixed costs that the order capacity-wise consumes times a parameter which represents the additional costs (rounded to quarters), i.e., $p_i^{new} = \frac{1}{4} round(4 \cdot 1.1 \cdot F\frac{q_i}{Q})$. These new costs are still slightly dependent on the distance since the fixed costs F is present in this formula and F is derived from the best known solution in the original problem with timing restrictions.

3.5 Test results

As is common practice in the literature, we run our heuristics 10 times and record the best and the average solutions. A summary of the results is displayed in Tables

3.5-3.8, while the detailed results can be found in Appendix B.2. For all methods, unless stated otherwise below, “best” corresponds to the average gap of the best found solution from the ten runs compared to the best known solution (BKS), “avg” corresponds to the average gap of the average solution, and “#BKS” corresponds to the number of times the method retrieved a best known solution. A few notes on the heuristics in the literature. For some heuristics only the best found solution is reported while only one run is provided of the RIP (randomized construction-improvement-perturbation, Bolduc et al. (2008)). There is a small issue concerning truncated coordinates in the heuristics of Côté and Potvin (2009) and Potvin and Naud (2011), which implies that only a part of their results can be compared. It is unclear whether this truncation is positive or negative for their heuristic. Another interesting note is that the Tabu Search (TS) of Potvin and Naud (2011) uses the same code as in Côté and Potvin (2009) (TS25), but now with 50,000 iterations instead of the 25,000 as in Côté and Potvin (2009). Although the number of iterations has been doubled, the results of the TS in Potvin and Naud (2011) are worse. No explanation why these results are so different is given in Potvin and Naud (2011). The final note is that Kratica et al. (2012) reported the best result found over 20 runs of their GA (genetic algorithm) instead of the commonly used 10 and that the values of the AVNS-RN (adaptable variable neighborhood search-random neighborhood ordering) are reverse engineered since Stenger, Schneider, and Goeke (2013) only provides the current best-known solution and a gap. Therefore, the exact values of the AVNS-RN in Table B.2 could be slightly different, but the influence on the gaps is negligible. The other heuristics in Table 3.5 are the TS+ (tabu search with ejection chains, Potvin and Naud (2011)), AVNS (adaptable variable neighborhood search, Stenger, Vigo, Enz, and Schwind (2013)), MS-LS (multistart local-improvement search, Vidal et al. (2015)), MS-ILS (multistart iterative local search, Vidal et al. (2015)) and the UHGS (unified hybrid genetic search, Vidal et al. (2015)).

		RIP	TS25	TS	TS+	GA	AVNS	AVNS-RN	MS-LS	MS-ILS	UHGS	R-TS	LNS-F	LNS-S
CE (#14)	best	1.09%	0.19%	0.45%	0.35%	*4.42%*	0.23%	0.16%	1.44%	0.10%	0.04%	0.25%	0.08%	0.04%
	avg	-	0.43%	-	-	-	-	0.45%	2.73%	0.34%	0.17%	0.81%	0.32%	0.22%
	#BKS	0	5	4	4	*0*	2	7	0	8	10	4	8	10
G (#20)	best	2.38%	*0.61%*	*0.79%*	*0.76%*	*6.61%*	1.02%	1.02%	3.03%	0.85%	0.49%	1.05%	0.41%	0.24%
	avg	-	*0.87%*	-	-	-	-	1.56%	3.93%	1.39%	0.80%	1.82%	0.77%	0.60%
	#BKS	0	*0*	*0*	*0*	*0*	0	0	0	0	2	0	1	2
Avg	best	1.85%	*0.34%*	*0.58%*	*0.50%*	*5.71%*	0.69%	0.67%	2.38%	0.54%	0.31%	0.72%	0.27%	0.16%
	avg	-	*0.59%*	-	-	-	-	1.11%	3.43%	0.95%	0.54%	1.40%	0.59%	0.44%

Table 3.5: Comparison on homogenous instances and original outsource costs.

The gaps of the heuristics compared to the best known solution (BKS) appear to be small in Table 3.5. However, this is deceiving since by construction around half of the costs consists of the fixed route costs. This is part of the reason why a relatively simple heuristic, like the R-TS, has only a gap of 0.72%, averaged over the instances. The LNS-Slow has on average a gap of 0.16%, while the UHGS has 0.31%, and the LNS-Fast has 0.27%. For the average solutions the gaps are closer to each other, 0.44% for the LNS-Slow against 0.54% for the UHGS and 0.59% for the LNS-Fast. What is interesting is that several of the BKS's in the CE set are found by many heuristics, including the simple R-TS heuristic. The differences are larger when we look at the larger, and seemingly more difficult G-instances. For the G-instances, the R-TS has a gap of 1.05%, the UHGS has a gap of 0.49%, and the LNS-Fast and the LNS-Slow have 0.41% and 0.24%, respectively. More importantly, the maximum gap of the best solution is relatively large for the UHGS, 2.03% (at G-12), while the largest gap of LNS-Fast is 0.86% (at G-07), and finally, for the LNS-Slow it is only 0.62% (at G-04). From the detailed results in B.2 we obtain that the LNS-Slow outperforms the UHGS on the G-instances with many orders per route (G-01 up to G-12), while the UHGS is only slightly better at the instances with many routes. This implies that our heuristics are better than the UHGS, which was the best heuristic for the VRPO before ours. We have retrieved all previous best known solutions and even improved the best known solution of 3 (of the 14) CE-instances and 18 (of the 20) G-instances. The results on our LNS's show that using different neighborhoods can be worthwhile since our LNS-Fast is much better than the AVNS(-RN). Our LNS-Fast and LNS-Slow have much better results than the MS-ILS. Since the MS-ILS already finds the best known solution on 125 of the 130 instances of Archetti et al. (2013) and performs better than the heuristics of Archetti et al. (2009), we think that our heuristics will perform well on those instances. The calculation time of the LNS-Fast is slightly less than the UHGS, 24.2 versus 26.4 minutes, while the calculation time of the LNS-Slow is three times more (77.6 minutes).

In the heterogenous case, which uses multiple types of trucks instead of one, the results are more pronounced. The LNS-Slow and LNS-Fast have a gap of 0.25% and 0.62%, respectively. We find 10 (out of 14) new best known solutions for the CE-instances and 20 (out of 20) for the G-instances. Furthermore, we have retrieved all the best known solutions except for CE-H-14, for which we found 1907.75 while the BKS is 1907.74. Again, the LNS-Slow is much better than the other heuristics.

		RIP	TS	TS+	GA	R-TS	LNS-F	LNS-S
CE (#14)	best	0.90%	0.65%	0.54%	*3.91%*	0.80%	0.38%	0.11%
	avg	-	-	-	-	1.64%	1.06%	0.54%
	#BKS	0	1	1	*0*	1	2	5
G (#20)	best	2.41%	*1.95%*	*0.97%*	*7.08%*	1.83%	0.78%	0.35%
	avg	-	-	-	-	2.69%	1.37%	0.87%
	#BKS	0	*0*	*1*	*0*	0	1	2
Avg	best	1.79%	*1.16%*	*0.71%*	*5.77%*	1.41%	0.62%	0.25%
	avg	-	-	-	-	2.26%	1.24%	0.73%

Table 3.6: Comparison on heterogenous instances and original outsource costs.

Table 3.7 shows the summary of the results of our three heuristics on the instances with half-original pricing and Table 3.8 on the instances with the new pricing. The results on these instances are in line with what is found on the original instances, namely, that the G-instances seem to be harder than the CE-instances and that the LNS-Slow is the best heuristic.

		homogenous			heterogenous		
		R-TS	LNS-F	LNS-S	R-TS	LNS-F	LNS-S
CE (#14)	best	0.19%	0.06%	0.00%	0.68%	0.10%	0.03%
	avg	0.63%	0.18%	0.10%	1.86%	0.36%	0.25%
	#BKS	8	10	12	2	8	11
G (#20)	best	1.05%	0.11%	0.04%	1.51%	0.19%	0.04%
	avg	1.56%	0.50%	0.33%	2.38%	0.70%	0.44%
	#BKS	0	4	13	0	4	15
Avg	best	0.70%	0.09%	0.03%	1.19%	0.15%	0.03%
	avg	1.19%	0.37%	0.24%	2.20%	0.57%	0.37%

Table 3.7: Comparison on half-original outsource costs.

		homogenous			heterogenous		
		R-TS	LNS-F	LNS-S	R-TS	LNS-F	LNS-S
CE (#14)	best	0.24%	0.07%	0.02%	0.38%	0.12%	0.02%
	avg	0.74%	0.25%	0.10%	0.91%	0.36%	0.23%
	#BKS	3	8	10	1	4	11
G (#20)	best	0.48%	0.16%	0.01%	0.66%	0.12%	0.01%
	avg	0.86%	0.50%	0.21%	1.08%	0.42%	0.23%
	#BKS	0	0	18	0	8	16
Avg	best	0.39%	0.13%	0.01%	0.55%	0.12%	0.01%
	avg	0.83%	0.37%	0.17%	1.02%	0.40%	0.23%

Table 3.8: Comparison on new outsource costs.

Additionally, we have tested our heuristics on the classical VRP (without outsourcing). To be more precise, we have used the CE-instances of Christofides et al.

(1979) and the tai-instances of Taillard (1993) for this test. The objective for these instances is to minimize the total distance without minimizing the routes. The results are presented in Table B.19. From this table we obtain that the gaps are 1.73% for the R-TS, 0.44% for the LNS-Fast, and 0.34% for the LNS-Slow. However, most of the moves of the LNS's put as much orders inside a route as possible, which is not necessary a good move for this problem. We also have tested our heuristics when minimizing the number of routes is the main objective and minimizing distance the secondary objective. The results are more pronounced. In other words, the differences between our heuristics are larger, but they give no additional insight and are therefore omitted. Surprisingly, the LNS-Slow did find, when minimizing the routes and then distances, two solutions which use less distance than the previous BKS's when only minimizing the distance. These results are bolded in Table B.19.

Finally, in Table 3.9, we present the results of the different moves used in the LNS-Slow. For each move we report the total relative improvement that the move found in ‰(TRI‰), which is the amount of improvement divided by the costs before the move, and the number of times the move found an improvement ($\#Impr$). These numbers are the averages over the ten runs and the averages over the instances with the same outsourcing costs structure (Original, Half-Original, and New). This table shows that the destroy move, compared to the other moves, is not promising since it finds on average an improvement 0.6 times. In other words, on many runs and many instances this move does almost nothing. The create and split moves are not so useful in the original pricing, which is not strange since (almost always) all routes are used. From the table we also obtain that all the other moves, both the ones from the literature and the new moves, are important. Most moves find an improvement at the beginning of the run. However, since these results are averaged, it is likely that each move finds good improvements at the beginning. We did not investigate whether a move influenced the remaining moves. For example, it could be that a move that has only a slight improvement brings the solution to a new neighborhood from which the other moves find better solutions. Furthermore, it could be that removing a move will not influence the outcomes of the heuristic. However, comparing our results to the results of the other LNS's indicates that different moves are beneficial.

Moves		Original		Half-Original		New		Avg	
#	Name	TRI‰	#Impr	TRI‰	#Impr	TRI‰	#Impr	TRI‰	#Impr
1	Cyclic(small, one-way)	5.6	9.7	4.6	8.6	3.5	8.4	4.6	8.9
2	Cyclic(small)	4.5	8.5	5.0	7.5	3.7	7.7	4.4	7.9
3	Cyclic(medium)	4.4	4.9	3.0	4.0	3.2	4.4	3.5	4.4
4	Cyclic(large)	4.0	2.9	2.9	2.4	2.1	2.7	3.0	2.7
5	Cyclic(small, candidate)	4.9	8.7	3.3	7.7	4.2	8.7	4.1	8.4
6	Cyclic(medium, candidate)	4.5	7.6	4.1	6.9	3.5	7.6	4.0	7.4
7	Cyclic(large, candidate)	3.8	3.3	3.7	3.3	2.8	3.5	3.4	3.4
8	Create + shifting	1.4	1.3	2.9	3.5	5.0	1.5	3.1	2.1
9	Destroy	1.8	0.7	1.4	0.6	1.0	0.6	1.4	0.6
10	Split + shifting	2.5	1.0	2.4	2.3	6.4	1.6	3.8	1.7
11	Bomb(small) + shifting	8.0	6.5	4.8	7.0	5.3	6.3	6.1	6.6
12	Bomb (large) + shifting	5.6	2.8	3.3	2.8	5.3	3.6	4.7	3.1
13	Ruin-and-repair(small)	4.8	7.6	4.1	7.6	4.2	8.7	4.4	7.9
14	Ruin-and-repair(large)	3.9	5.4	2.8	5.1	3.5	5.9	3.4	5.5
15	Reseed(small) + shifting	6.2	4.0	6.6	4.3	3.5	3.5	5.5	3.9
16	Reseed(large) + shifting	6.0	1.7	4.0	2.5	3.0	2.0	4.3	2.1
17	Tabu	4.2	9.1	3.4	7.7	4.2	10.5	3.9	9.1

Table 3.9: The value of the moves of the LNS-Slow (TRI = total relative improvement).

3.6 Conclusions and recommendations

We argue that the original test instances for the VRPO have as characteristic that larger orders are outsourced out of necessity. To get more diverse instances, we create two new pricing mechanisms which in turn yields new instances. To tackle the different instances we introduce new large neighborhood search heuristics. Our heuristics use several different ruin-and-repair moves. Most of these moves are new, including an important repair move. The computational results show that both our LNS's find on average better solutions than the previous best heuristic. Additionally, we have improved most best known solutions. Moreover, the results indicate that the new moves are worthwhile and that the LNS's are able to find good solutions on multiple types of instances. What is surprising is that these results are obtained without optimizing the different parameters but by randomly drawing the parameters within reasonable bounds.

We did not compare the heuristics on the running times. This is a deliberate choice since time is not necessarily a good measure. Not only is the necessary information missing to do a comparison, for example, how many cores were used, but also optimization of the code plays an important role. On the other hand, looking at only solution values or the number of iterations is not a good measure either. An idea of a good measure could be the number of basic operations together with the memory usage.

Chapter 4

Estimations for the VRP with Order outsourcing

4.1 Introduction

This chapter, which is based on Huijink, Kant, and Peeters (2015), researches another method to determine which orders to outsource. In the previous chapter, heuristics are used. Not every routing software package or transport management system is able to determine which orders to outsource. Additionally, there is not much time to decide which orders to outsource since the orders need to be reallocated. It could well be that even heuristics take too much time. Finally, determining routes for the remaining orders is not so useful since the company will receive orders that are outsourced to it. Therefore, the companies are interested in so-called "rules of thumb" to determine which orders to outsource. These rules of thumb should be fast and simple.

Whether or not an order should be outsourced depends on the (expected) routing costs of the order when the company delivers this order by itself and its outsourcing costs. However, the (expected) routing costs of the order depends on other orders which are not outsourced. This is the case for all orders, which creates an interdependence between them. Furthermore, it could be that an order is outsourced even when the expected routing costs of that order are smaller than the outsource costs. For example due to the capacity constraints or because other orders are more profitable to deliver. For the classical VRP, there exist several methods to estimate the routing costs (of an order) (Daganzo (1984); Fleischmann (1998); Goudvis (2001); Figliozzi (2009); Baghuis (2014)). Each of these methods splits the estimation into three separate parts, the fixed, the stem and the inter-order part. The fixed part estimates the fixed costs for delivering an order. The stem estimates the costs for driving to the area in which the order should be delivered. The inter-order part estimates the costs for driving between orders. The estimations use the characteristics of the order and the amount of orders in the area or the characteristics of orders that are near to the order. The possibility of outsourcing orders has as a consequence that the number of routes that one should use, which is a vital part in the estimations of the VRP, is unknown.

To the best of our knowledge, the paper of Hall and Racer (1995) is the first one that estimates which orders to outsource, in other words, uses estimations in the VRPO. Hall and Racer (1995) use the method of Daganzo (1984), which is an estimation method for the VRP, to estimate the delivery costs of an order. The orders which have the highest, and non-negative, unit regret, which is the outsource costs minus the estimated delivery costs divided by the demand, are selected to be delivered by the own fleet. Baghuis (2014) presents another method to estimate which orders to outsource. This method estimates the inter-order costs by looking at the direct distance between an order and its neighboring orders. The selection method of Baghuis (2014) selects batches of orders such that the total demand of the orders in each batch is more or less the capacity of a truck. Another estimation, which is used by several heuristics for the VRPO (Côté and Potvin (2009); Stenger, Schneider, and Goeke (2013)), is to assume that the estimated delivery costs are zero and use the same selection as Hall and Racer (1995).

In this chapter, we present a new estimation method. This method estimates the inter-order costs by looking at the insertion costs of the order and its neighboring orders. Together with this estimation method, we create a new selection method. Additionally, we also adopt the methods of Fleischmann (1998) and Goudvis (2001), which originally are for the VRP, to give an estimation method for the VRPO. To measure the quality of the different estimation methods we develop three tests and use these to compare the different estimations for the VRPO. Finally, we also compare the estimations on the classical VRP.

The rest of this chapter is organized as follows. The different estimation methods are introduced and discussed in Section 4.2, the tests in Section 4.3, and the results in Section 4.4. The conclusions and recommendations are presented at the end of this chapter.

4.2 Estimations: an overview

Each of the estimations has a two step procedure. In the first step a score is assigned to and in the second step this score is used to select the orders that will be delivered. Each method assigns to each order a score which can be written as $s_i = \frac{p_i - EstCosts_i}{q_i}$ where p_i is the *outsource costs* for the order, $EstCosts_i$ is the *estimated costs* for delivering the order instead of outsourcing it, q_i is the *demand* of the order and Q is the *capacity* restriction of the route. In other words, the score is the regret for not delivering the order per unit demand. For most estimations, the estimated costs for delivering consists of the sum of the *fixed* (F_i), which estimates and allocates the fixed costs of a route over the orders in a route, *stem* (S_i), which estimates and allocates the costs for driving to the area in which the orders are located, and *inter-order* cost (IO_i), which estimates and allocates the costs for driving from the location of one order to another. This implies that the estimation does not assign the marginal or additional costs of the order, but it estimates the total costs that are assigned to the order. This gives the following basic algorithm.

Algorithm 4.2.1 *Basis*

Step 1: Assigning a score

1: Estimate the costs of delivering, $EstCosts_i = F_i + S_i + IO_i$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Select which orders to deliver based on the ranked scores and a stopping criterium

The rank of order i is denoted by $\pi(i)$ where the inverse $\pi^{-1}(r)$ denotes the order with rank r. Furthermore, the distance between orders i and j is denoted by d_{ij}, where order 0 denotes the depot.

4.2.1 Initialization estimation

Several of the heuristics developed to address the VRPO use a very simple selection method (Côté and Potvin (2009); Potvin and Naud (2011); Stenger, Schneider, and Goeke (2013); Stenger, Vigo, Enz, and Schwind (2013)). The main idea of their initialization, which is also an estimation method, is to ignore the delivery costs in step 1. In step 2 the orders are selected until the available capacity runs out. To be more precise, $EstCosts_i = 0$, which gives the following algorithm.

Algorithm 4.2.2 *Initialization estimation*

Step 1: Assigning a score

1: Estimate the costs of delivering, $EstCosts_i = 0$

2: Assign a score to each order, $s_i = \frac{p_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Select the orders $\pi^{-1}(1), ..., \pi^{-1}(k)$ *to deliver and outsource the orders* $\pi^{-1}(k+1), ..., \pi^{-1}(n)$*, where* k *is such that* $\sum_{\ell=1}^{k+1} q_{\pi^{-1}(\ell)} > m \cdot Q \geq \sum_{\ell=1}^{k} q_{\pi^{-1}(\ell)}$

4.2.2 Hall estimation

In step 1, Hall and Racer (1995) use the method of Daganzo (1984) to estimate the delivery costs of an order. The estimation of Daganzo (1984) assumes that the orders are randomly distributed over the area and that the orders are homogenous. Due to these assumptions the inter-order costs are equal for each order. The inter-order costs for each order are estimated using the estimation for the traveling salesman problem. The total distance travelled in the traveling salesman problem is estimated by $k\sqrt{n \cdot A}$ where k is a constant depending on the metric, n is the number of orders and A is the size of the delivery area (Daganzo (1984)). However, in the VRPO it is not known beforehand which or how many orders are delivered. Therefore, Hall and Racer (1995) assume that x orders will be delivered and calculate the estimation for each possible value of x. The fixed costs are estimated by assigning the capacity fraction of the fixed route costs to the order, $F\frac{q_i}{Q}$. The stem costs are estimated by a capacity fraction of the costs to drive to the area and back, $2v \cdot d_{0i}\frac{q_i}{Q}$ where v is the costs per kilometer of a truck, and the inter-order costs are the first Taylor approximation of the traveling salesman problem estimation, which depend on the number of orders x, $v\frac{k}{2\sqrt{\frac{x}{A}}}$. For each value of x, step 2 consists of selecting the orders with a positive score while taking the total capacity into account. This gives the following algorithm for a given value x.

Algorithm 4.2.3 *Hall estimation for given value x*

Step 1: Assigning a score

1: Estimate the costs of delivering, $EstCosts_i = F\frac{q_i}{Q} + 2v \cdot d_{0i}\frac{q_i}{Q} + v\frac{k}{2\sqrt{\frac{x}{A}}}$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Select the orders $\pi^{-1}(1), ..., \pi^{-1}(k)$ *to deliver and outsource the orders* $\pi^{-1}(k+1), ..., \pi^{-1}(n)$*, where* $k = \min\{r, t\}$, $\sum_{\ell=1}^{r+1} q_{\pi^{-1}(\ell)} > m \cdot Q \geq \sum_{\ell=1}^{r} q_{\pi^{-1}(\ell)}$ *and* $s_{\pi^{-1}(t)} \geq 0 > s_{\pi^{-1}(t+1)}$

This estimation depends on the number of orders x that were assumed to be delivered. Note that for a given x the number of orders delivered is given by $\pi^{-1}(k)$ which is not necessary equal to x. The final estimation is the estimation for which the number of orders delivered, $\pi^{-1}(k)$, is equal to the number of orders assumed to be delivered, x. In the case that there are multiple x's that satisfy this condition, then the one which has the minimal estimated cost is chosen.

4.2.3 Baghuis and Baghuis(Adjusted) estimation

Another estimation is developed by Baghuis (2014). For step 1, Baghuis (2014) uses the distance between the depot and the orders as a proxy for the stem costs and the distance between orders as a proxy for the inter-order costs. Using results from several test-instances, Baghuis (2014) estimates the stem costs of a route, which is denoted by S^B and is not the stem costs of an order, by taking the average distance between the depot and the 30% closest orders. Similarly, the inter-order costs of an order $(v \cdot IO_i^B)$ are estimated by taking the average distance between the order and its 5 closest orders. Additionally, Baghuis (2014) uses the undershoot, which is the part of the capacity that is not used in a route and has its roots in inventory management, to estimate the costs. The fixed costs are estimated by $F\frac{q_i}{Q-U}$, where the undershoot is given by $U = \frac{\sum_{i \in N} q_i^2}{2\sum_{i \in N} q_i} - \frac{1}{2}$. Baghuis (2014) estimates the stem costs of an order by $v\frac{d_{0i}}{\bar{d}_0}\frac{\bar{q}}{Q}S^B$ where $\bar{d}_0$ is the average distance from the depot to an order, $\bar{q}$ is the average demand of an order. In other words, the stem costs of an order are the average stem costs of an order multiplied by the ratio of the distance from the depot to the location of the order and the average distance of the locations of the orders to the depot. For step 2, the ranking is divided into disjoint batches, each with the most orders such that the demand of the orders is less than the capacity of the routes. All batches with positive score are delivered.

Algorithm 4.2.4 *Baghuis estimation*

Step 1: Assigning a score

1: Estimate the costs of delivering, $EstCosts_i = F\frac{q_i}{Q-U} + v\frac{d_{0i}}{d_0}\frac{\bar{q}}{Q}S^B + v \cdot IO_i^B$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Divide the ranking into batches $(\pi^{-1}(1), ..., \pi^{-1}(u_1)), (\pi^{-1}(u_1+1), ..., \pi^{-1}(u_2)), ...,$ *where for every* i *it holds that* $\sum_{\ell=u_{i-1}+1}^{u_i+1} q_{\pi^{-1}(\ell)} > Q \geq \sum_{\ell=u_{i-1}+1}^{u_i} q_{\pi^{-1}(\ell)}$. *Select the orders* $\pi^{-1}(1), ..., \pi^{-1}(u_k)$ *to deliver and outsource the orders* $\pi^{-1}(u_k+1), ..., \pi^{-1}(n)$, *where* $k = \min\{r, m\}$, $\sum_{\ell=1}^{u_r} s_{\pi^{-1}(\ell)} > \sum_{\ell=1}^{u_{r+1}} s_{\pi^{-1}(\ell)}$ *and* m *is the number of available trucks*

Hall and Racer (1995) select the orders until their score is negative, which can imply that a route is not fully utilized. However, the Hall estimation assumes that a route is fully utilized, which can lead to an underestimation of the costs. To avoid this, Baghuis (2014) uses an undershoot and selects the orders in batches. However, the selection of Baghuis (2014) can result in not fully utilized routes. Often one can utilize a route fully by allowing a small detour. Therefore, we use overlapping batches instead of disjoint batches where each batch contains the previous batch and the demand in each batch is increased by the capacity of a route. This gives the following algorithm.

Algorithm 4.2.5 *BaghuisAdjusted estimation*

Step 1: Assigning a score

1: Estimate the costs of delivering, $EstCosts_i = F\frac{q_i}{Q-U} + v\frac{d_{0i}}{d_0}\frac{\bar{q}}{Q}S^B + v \cdot IO_i^B$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Divide the ranking into batches $(\pi^{-1}(1), ..., \pi^{-1}(u_1)), (\pi^{-1}(1), ..., \pi^{-1}(u_2)), ...,$ *where for every* i *it holds that* $\sum_{\ell=1}^{u_i+1} q_{\pi^{-1}(\ell)} > i \cdot Q \geq \sum_{\ell=1}^{u_i} q_{\pi^{-1}(\ell)}$. *Select the orders* $\pi^{-1}(1), ..., \pi^{-1}(u_k)$ *to deliver and outsource the orders* $\pi^{-1}(u_k+1), ..., \pi^{-1}(n)$, *where* $k = \min\{r, m\}$, $\sum_{\ell=1}^{u_r} s_{\pi^{-1}(\ell)} > \sum_{\ell=1}^{u_{r+1}} s_{\pi^{-1}(\ell)}$ *and* m *is the number of available trucks*

4.2.4 Fleischmann estimation

In step 1, we use the estimation of Fleischmann (1998), which bears some resemblance to the estimation of Daganzo (1984). Fleischmann (1998) assumes that orders with similar characteristics are situated on a ring around the depot instead of a random uniform distribution. A main part of the model is to determine the distance between two orders in the ring. This distance (and its costs) can be estimated using solutions from a part of the instances, but we choose to use an adjusted model of Daganzo (1984) to estimate the inter-order costs. We choose a circle with as center the order for which we want to estimate the delivery costs, such that the demand of the all the orders inside the circle is at least the capacity of a route. Then, the inter-order costs are given by $IO_i = v(1 - \frac{q_i}{Q})\frac{k}{2\sqrt{\frac{y_i}{B}}}$ where B is the area of the circle and y_i is the number of orders in the circle. Just as in Fleischmann (1998), we have the fraction $(1 - \frac{q_i}{Q})$ since there are $y_i - 1$ inter-order distances and not y_i as in Hall and Racer (1995). The main difference between the Hall and Fleischmann method is that the Fleischmann method uses orders close to the order to estimate the inter-order costs instead of all the orders as in Hall. In step 2, we use our selection method which we also use in the BaghuisAdjusted method. This gives the following algorithm.

Algorithm 4.2.6 *Fleischmann estimation*

Step 1: Assigning a score

1: Estimate the costs of delivering, $EstCosts_i = F\frac{q_i}{Q} + 2v \cdot d_{0i}\frac{q_i}{Q} + v(1 - \frac{q_i}{Q})\frac{k}{2\sqrt{\frac{y_i}{B}}}$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Divide the ranking into batches $(\pi^{-1}(1), ..., \pi^{-1}(u_1)), (\pi^{-1}(1), ..., \pi^{-1}(u_2)), ...,$ *where for every* i *it holds that* $\sum_{\ell=1}^{u_i+1} q_{\pi^{-1}(\ell)} > i \cdot Q \geq \sum_{\ell=1}^{u_i} q_{\pi^{-1}(\ell)}$. *Select the orders* $\pi^{-1}(1), ..., \pi^{-1}(u_k)$ *to deliver and outsource the orders* $\pi^{-1}(u_k+1), ..., \pi^{-1}(n)$, *where* $k = \min\{r, m\}$, $\sum_{\ell=1}^{u_r} s_{\pi^{-1}(\ell)} > \sum_{\ell=1}^{u_{r+1}} s_{\pi^{-1}(\ell)}$ *and* m *is the number of available trucks*

4.2.5 Goudvis estimation

For part 1, Goudvis (2001) estimates the inter-order costs by looking at the direct distance between close orders. Unlike Baghuis (2014), Goudvis (2001) assigns a weight to each order and uses a weighted average. However, the only part of the weight of Goudvis (2001) which has any meaning in the VRPO (or the normal VRP) is that it depends on the distance between the location of order i, which is the order of interest, and the location of another order j, in other words, $\frac{1}{1+d_{ij}}$. The other parts, for example whether the time-windows of the orders are compatible, have to be ignored. Since only profitable orders are delivered, we use the profit for delivering an order as an additional part in the weight, which gives the weight $Pr_j(i) = \frac{1}{1+d_{ij}} \frac{p_j - d_{ij}}{q_j}$. As estimation of the inter-order distance IO_i^G, we take the order j with the highest score $Pr_j(i)$ and estimate the distance by the distance between the orders i and j (d_{ij}). The remaining estimations are the same as in the Hall estimation. Part 2 is the same as in the Fleischmann estimation, which gives the following algorithm.

Algorithm 4.2.7 *Goudvis estimation*

Step 1: Assigning a score

1: Estimate the costs of delivering,

$$EstCosts_i = F\frac{q_i}{Q} + 2v \cdot d_{0i}\frac{q_i}{Q} + v \cdot IO_i^G$$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Divide the ranking into batches $(\pi^{-1}(1),...,\pi^{-1}(u_1)),(\pi^{-1}(1),...,\pi^{-1}(u_2)),...,$ *where for every* i *it holds that* $\sum_{\ell=1}^{u_i+1} q_{\pi^{-1}(\ell)} > i\cdot Q \geq \sum_{\ell=1}^{u_i} q_{\pi^{-1}(\ell)}$*. Select the orders* $\pi^{-1}(1),...,\pi^{-1}(u_k)$ *to deliver and outsource the orders* $\pi^{-1}(u_k+1),...,\pi^{-1}(n)$*, where* $k=\min\{r,m\}$*,* $\sum_{\ell=1}^{u_r} s_{\pi^{-1}(\ell)} > \sum_{\ell=1}^{u_{r+1}} s_{\pi^{-1}(\ell)}$ *and* m *is the number of available trucks*

4.2.6 Huijink and Huijink(10) estimation

Both Goudvis (2001) and Baghuis (2014) use the distance between the locations of two orders to estimate the inter-order costs. Another method to estimate the inter-order costs is to look at the insertion costs of an order. First, for each order i, we take the 5 orders which have the locations closest to the location of i. For each pair ($\{k,l\}$) in the set of pairs that we can make with those 5 orders ($Pairs(i)$), we calculate the insertion distance of the order i. Since an order is more likely to be inserted when the insertion distance is small, we take the weighted average. However, we cannot use the insertion distance as a weight since it could be that this is equal to zero, hence we have to use $1+insertiondistance$ as a weight. This gives the following estimation of the inter-order costs,

$$IO_i^H = \frac{1}{\sum_{\{k,l\}\in Pairs(i)} \frac{1}{1+d_{ki}+d_{il}-d_{kl}}} \sum_{\{k,l\}\in Pairs(i)} \frac{d_{ki}+d_{il}-d_{kl}}{1+d_{ki}+d_{il}-d_{kl}}.$$

The other estimations are the same as in the Fleischmann method. Part 2 is also the same as in the Fleischmann estimation, which gives the following algorithm.

Algorithm 4.2.8 *Huijink estimation*

Step 1: Assigning a score

1: Estimate the costs of delivering,

$$EstCosts_i = F\frac{q_i}{Q} + 2v\cdot d_{0i}\frac{q_i}{Q} + v\cdot IO_i^H$$

2: Assign a score to each order, $s_i = \frac{p_i - EstCosts_i}{q_i}$

Step 2: Selecting which orders to deliver

3: Rank the orders in a descending manner according to s_i

4: Divide the ranking into batches $(\pi^{-1}(1), ..., \pi^{-1}(u_1)), (\pi^{-1}(1), ..., \pi^{-1}(u_2)), ...,$ *where for every* i *it holds that* $\sum_{\ell=1}^{u_i+1} q_{\pi^{-1}(\ell)} > i \cdot Q \geq \sum_{\ell=1}^{u_i} q_{\pi^{-1}(\ell)}$. *Select the orders* $\pi^{-1}(1), ..., \pi^{-1}(u_k)$ *to deliver and outsource the orders* $\pi^{-1}(u_k+1), ..., \pi^{-1}(n)$, *where* $k = \min\{r, m\}$, $\sum_{\ell=1}^{u_r} s_{\pi^{-1}(\ell)} > \sum_{\ell=1}^{u_{r+1}} s_{\pi^{-1}(\ell)}$ *and* m *is the number of available trucks*

For the Huijink10 method, we use the 10 closest orders instead of 5 (which increases the number of pairs from 10 to 45).

4.3 Tests

We develop three tests to measure the quality of each estimation method. The most important test is quality of the solution, which is tested by comparing the costs of the best found solution, which uses the outsourcing decision of the estimation method, and the costs of the best known solution. The second test is correctness of the orders outsourced (delivered). In other words, does the estimation outsource (deliver) the same orders as the best known solution? This is tested by comparing the number of orders that are outsourced (delivered) by both the estimation and the best known solution and the number of orders that are outsourced (delivered) by either the method or the best known solution. The final test is the quality of the estimation, which is tested by comparing the estimated costs and the costs of the best known solution. If one can estimate the costs of an instance quite reasonably, then one can quickly determine what the influence would be if the outsource costs would change. The tests are formalized in Table 4.1, where we use the following notation. The *Costs* of the method are the best found solution of the ten runs of the AVNS-Fast of Chapter 3, where the outsource costs are set such that the orders that were outsourced (delivered) by the estimation method are outsourced (delivered) by the AVNS-Fast. We choose the AVNS-Fast, since it has a gap of less than 0.10% on the CE-instances. The *BKSCosts* are the costs of the best known solution of the instance. *Both* are the number of orders that are outsourced (delivered) by both the estimation method and the BKS and *Either* are the number of orders outsourced (delivered) by either the method or the BKS. Finally, the *EstCosts* are the estimated costs of the estimation method. Note that the quality of the solution is the most important test for the coalition in the pricing based structure (Section 2.4). Therefore, we will focus on that test when drawing conclusions.

Name	*test*
Quality of solution	$\frac{Costs - BKSCosts}{BKSCosts} * 100\%$
Correctness of outsourced (delivered)	$\frac{Both}{Either} * 100\%$
Quality of estimation	$\frac{\lvert EstCosts - BKSCosts \rvert}{BKSCosts} * 100\%$

Table 4.1: The tests.

4.4 Test results

The detailed results of the computational experiments can be found in Appendix C. Table 4.2 presents the summarized results of the quality of the solution test.

Instance set	Init	Hall	Bag	Fleisch	Goud	Huij	BagAdj	Huij10
CE-Orig	0.61	0.51	2.65	0.56	0.53	0.60	0.73	0.74
CE-Half-Orig	2.46	2.35	2.87	1.64	2.61	1.98	2.20	1.88
CE-New	3.03	3.10	3.86	2.64	4.19	1.46	2.16	1.67
Average	2.04	1.99	3.13	1.61	2.44	1.35	1.70	1.43

Table 4.2: The average quality of solution for each method in %.

The gaps are due to the choice of the method on which orders to outsource, in other words, the gap is close to zero when the heuristic is allowed to choose which orders to outsource. Of the methods in the literature, the simple initialization is better than both the Hall and Baghuis method. From the methods we translated to the VRPO (Fleischmann and Goudvis), the Goudvis method performs worse than the Initialization and the Fleischmann method performs better than the Hall method. One could conclude from the fact that the Fleischmann method is better than the Hall method that it is better to determine the inter-order distance while ignoring the fact that some orders would not be delivered than assuming the same inter-order distance for each order. Furthermore, the BaghuisAdj method performs much better than the original, which shows that our selection method outperforms the one of Baghuis (2014). What is interesting is that only changing the estimation for the inter-order costs already has a large influence on the results.

Both the Huijink and Huijink10 method are on average better than all the other methods. Furthermore, it shows that the Huijink method is quite robust with respect to the amount of neighbouring orders inspected. What is interesting is that almost all methods perform similarly well on the CE-Orig-instances, while there are large differences as well as larger gaps on the other instances. The gaps between outsourcing everything and the best known solutions is 129%, 28%, and 18% for the Orig, Half-Orig, and the New-instances, respectively. Furthermore, in the Half-Orig and New-instances there are some instances where using one route more (or less) than the number of routes in the best found solution does not have a large influence on the costs, for example, the instance Half-Orig CE-02 for which the best known solution uses 4 routes and a solution with 3 routes has a gap of just 0.14%. Note that this is never the case for the Orig-instances since the best known solution always use all the available routes in these instances. Hence, one would expect lower gaps at the Half-Orig and New-instances than on the Orig-instances. The Fleischmann method is the best performing one on the Half-Original-instances and the Huijink method on the New-instances. The reason that the gaps appear to be small is partly due to the fact that a large part of the costs comes from the fixed costs of the routes (on average 42%).

Instance set		Init	Hall	Bag	Fleisch	Goud	Huij	BagAdj	Huij10
CE-Orig	Delivered	97.30	97.61	95.30	97.20	97.10	96.68	96.66	97.04
	Outsourced	79.49	80.84	68.89	78.28	77.60	74.57	75.09	77.09
CE-Half-Orig	Delivered	73.05	63.24	79.85	81.35	77.84	81.55	72.75	81.69
	Outsourced	38.61	68.93	69.08	69.51	68.58	69.80	68.92	71.40
CE-New	Delivered	81.09	67.72	82.12	84.68	78.71	87.09	83.94	86.91
	Outsourced	15.35	30.64	40.66	37.83	41.66	48.45	45.25	48.82
Average	Delivered	83.81	76.19	85.76	87.74	84.55	88.44	84.45	88.55
	Outsourced	44.48	60.14	59.54	61.87	62.61	64.28	63.09	65.77

Table 4.3: The average correctness of outsourced (delivered) of each method in %.

Table 4.3 shows in conjunction with Table 4.2 that the second test, which tests the correctness of outsourcing, is rather weak. For example comparing the Huijink and the Fleischmann method on the CE-Half-Orig-instances shows that Huijink chooses on average more orders correctly, while Fleischmann has a lower gap. Similarly, although the Hall method has more orders correct than the Initialization, it does not show in the gap. This is largely due to the fact that the VRPO often has many solutions with nearly the same costs, even while the orders outsourced in each solution is quite different. Due to this grey area it could be that two solutions have almost the same costs, while they are completely different.

Instance set	Hall	Bag	Fleisch	Goud	Huij	BagAdj	Huij10
CE-Orig	4.36	4.69	7.64	4.01	6.70	5.10	4.48
CE-Half-Orig	3.92	4.49	7.45	1.64	6.59	4.51	4.57
CE-New	3.76	4.27	7.52	3.17	8.57	4.43	6.71
Average	4.01	4.48	7.54	2.94	7.29	4.68	5.25
Taillard (VRP)	47.99	*32.14*	12.51	25.82	10.98	32.40	9.76

Table 4.4: The average quality of estimation of each method in %.

Comparing the estimations of the costs on the VRPO instances shows that the Goudvis method estimates the best, closely followed by all but the Fleischmann method and our method. Our method which looks at the ten closest orders performs much better on the estimation part than our method which looks at the five closest orders. On the VRP instances the results are reversed, our methods and the Fleischmann method outperform all methods, even the Hall method, which is based on the estimation of Daganzo (1984). What is interesting is that all methods, except the Fleischmann method and our methods, have a gap in the VRP instances which is at least five times the gap of the VRPO instances. Furthermore, the original Baghuis method needs more trucks than available in three instances in the VRP. The reason for the discrepancy between the gaps is that the Fleischmann method, the Baghuis methods, as well as our methods, underestimate the distance costs (sum of inter-order and stem) of the VRP instances, while the other methods overestimate these costs. On the other hand, the Baghuis methods overestimate the fixed-route costs while the other methods slightly underestimate these costs. Hence, the under-estimation of the distance and the overestimation of the fixed-route costs partly cancel each other out. Similarly, the over-estimation of the distance and the under-estimation of the fixed routes partly cancel each other out. Another reason is that in the VRPO instances the routes have a large fixed cost, which leads to a minimization of routes and hence the distance is increased.

It is interesting that the method which estimates the total costs quite good, the Goudvis estimation, has a bad solution after the routing. Surprisingly, this method has on average roughly the same amount of orders correctly outsourced (delivered) as the other good performing methods. Therefore, one can conclude that the Goudvis method estimation is such that a few important orders are outsourced (delivered) while they should not. The results confirm our findings in Chapter 3 that the Orig instances have certain characteristics. It is interesting that ignoring the interdependence and using an estimation leads to a gap of at most two percent for the Huijink estimation. Finally, it is surprising how far off the estimations are in the VRP instances.

4.5 Conclusion and recommendations

The results on these instances indicate that the estimations are a viable option for determining which orders to outsource in the pricing based structure. Our estimation method scores the best on quality of solution, which is the most important measure for the pricing based structure. Surprisingly, the results on the quality of the solution show that a good estimation of the total costs not necessary results in good choices on which orders to outsource (deliver). One would expect that a better estimation would result in better costs after the outsourcing. However, since this is not the case, it deserves more attention. Furthermore, the estimations ignore the interdependence between the order when estimating which orders to outsource. One method to solve this is to use each method a few times where only the orders that are delivered are allowed to be used for estimating the delivery costs. Other follow up research is on other estimations, for example, using a minimal spanning tree together with matching to determine which orders to outsource. Finally, most estimations for the VRP have their roots in estimations for the Traveling Salesman Problem (TSP) (Daganzo (1984)). However, as far as we are aware of, there is no research on estimations of the TSP with profits, which is the TSP variant of the VRPO (Feillet et al. (2005)). Most of the methods we discussed can readily be adapted to estimate which orders to outsource in the TSP with profits.

Chapter 5

Pricing based structure: the pricing

5.1 Introduction

In this chapter, we analyze what the outsource costs should be for the pricing based structure. Where in the previous two chapters we assumed that the outsource costs where known and we looked at one company, we now look at the pricing based structure and try to find the best pricing mechanism. In the pricing based structure, companies agree on a pricing scheme, which consists of a fee for delivering the order and the inter-depot costs of the order, together with a division of the delivery area into regions. Each company is assigned to a region and is obliged to deliver orders outsourced in that area. However, there is no obligation to outsource orders. In principle, the outsource costs may differ per company and it may depend on various parameters. What the outsource costs should be depends on the objective of the companies. Here we assume that the companies want to minimize the joint costs, which is equivalent to maximizing the joint profit. Examples of other objectives are to maximize the absolute or relative profit of each company. Due to the underlying complexity, it is not easy to find the optimal parameters of the pricing scheme. Therefore, we resort to a surrogate model, namely, Kriging (Forrester et al. (2008)). A surrogate model approximates the outcome of parameter configurations using the actual outcomes of several configurations. Iteratively, the surrogate model is updated using the outcome of a promising configuration.

The outcome of a configuration, which is in this case the joint costs of cooperating, is calculated as follows. The LNS-Faster, which is a variant of the LNS-Fast (Chapter 3), determines which orders are outsourced. See Appendix D for the details of the LNS-Faster. The outsourced orders are transshipped and the LNS-Faster is used to calculate the costs after the outsourcing. Section 5.2 discusses the pricing model and how we determine the fee. In Section 5.3, we present the test instances together with the inter-depot costs. We report the results in Section 5.4 and the conclusion and recommendation at the end of this chapter.

5.2 Pricing model

The companies not only agree on a pricing mechanism for which the companies can outsource orders, but also on the region in which a company is obliged to deliver the outsourced orders. This pricing scheme, which consists of a fee for the company that delivers the order and the inter-depot costs of the order, and the regions are not determined on a daily basis, but are fixed for a longer period of time. We assume that the regions are already assigned and that the goal of the companies is to minimize the joint operational costs. However, in the pricing based structure the companies decide themselves in the operational phase which orders they outsource. To be able to make a good decision on which orders to outsource, the company needs to know the fee and inter-depot costs when outsourcing an order.

The set of companies that form a cooperation is denoted by $N = \{1, ..., n\}$. The outsource costs of an order from company $i \in N$ that has a location in the region of company $j \in N$ are given by the function $p_{ij}(.)$. The outsource costs $p_{ij}(.)$ consists of the fee that company i has to pay to company j, which is denoted by the function $f_{ij}(.)$, and the inter-depot costs for transhipping the order from company i to company j, which is denoted by the function $ID_{ij}(.)$. Hence, the costs to outsource an order from company i to company j, in which region the location of the order lies, are given by $p_{ij}(.) = f_{ij}(.) + ID_{ij}(.)$. For all companies $i \in N$, the vector of outsource costs of a company, $\mathbf{p}_i(.)$, is defined by $\mathbf{p}_i(.) = (p_{i1}(.), ..., p_{in}(.))$. Similarly, the vector of outsource costs, $\mathbf{p}(.)$, is defined by $\mathbf{p}(.) = (\mathbf{p}_1(.), ..., \mathbf{p}_n(.))$. Since the inter-depot transportation costs depend on all outsourced orders, there is an interdependence of the outsource costs. Each company decides, given the outsource costs $\mathbf{p}(.)$, which orders it outsources. The total outsource costs that company $i \in N$ pays is denoted by $P_i^{paid}(\mathbf{p}(.))$ and the fee that it receives is denoted by $P_i^{received}(\mathbf{p}(.))$. The routing costs of the company, which depends on the orders that it did not outsource together with the orders outsourced to it, are denoted by $VRP_i(\mathbf{p}(.))$. This gives that the total costs of company i, $Costs_i(p(.))$, is defined by $Costs_i(\mathbf{p}(.)) = P_i^{paid}(\mathbf{p}(.)) - P_i^{received}(\mathbf{p}(.)) + VRP_i(\mathbf{p}(.))$. The companies want to minimize the total costs, which gives the following model.

$$\min_{\mathbf{p}(.)} \sum_{i \in N} Costs_i(\mathbf{p}(.))$$
$$s.t.\ \ Costs_i(\mathbf{p}(.)) = P_i^{paid}(\mathbf{p}(.)) - P_i^{received}(\mathbf{p}(.)) + VRP_i(\mathbf{p}(.))\ \forall\ i \in N$$

Note that $P_i^{paid}(\mathbf{p}(.)) - P_i^{received}(\mathbf{p}(.))$ is equal to the inter-depot costs that company i pays. This implies that the companies minimize the total routing costs together with the inter-depot costs. The companies optimize the pricing scheme for a test period. This test period consists of several days and should be representative for the period for which they want to determine the pricing.

Due to the underlying complexity, we have to use a surrogate model to find the optimal outsource costs $\mathbf{p}(.)$. Kriging (Forrester et al. (2008)) is such a surrogate model, which for finding the best pricing mechanism operates as follows.

Algorithm 5.2.1 *The Kriging model*

Setup phase

Determine a set of initial pricing schemes (fees and inter-depot costs)

For every initial pricing scheme

Compute, given the pricing scheme, the total costs

Kriging

Repeat the following until a stopping criterion is met

Determine the surrogate model (Kriging parameters)

Determine the pricing scheme which is most promising according to the surrogate model

Compute the total costs of this pricing scheme

The first step in Kriging is to determine a few initial pricing schemes and calculate the costs of these. Then, Kriging determines, using maximum likelihood, what the outcomes would be of all possible pricing schemes. The outcomes of the Kriging model are searched for the most promising pricing scheme and the real costs of this scheme are calculated. This improves the accuracy of the model and we repeat the procedure until a stopping criterion is met. How we calculate, given the pricing scheme, the total costs is presented in Algorithm 5.2.2. To avoid the interdependence of the inter-depot costs and the outsource costs, we initially estimate the inter-depot costs for any order. If necessary, we adjust the inter-depot costs and recalculate which orders the companies outsource.

Algorithm 5.2.2 *Calculation, given the pricing scheme, of the costs*

Determine which orders to outsource and the inter-depot costs

Repeat the following until the inter-depot costs are estimated 'correctly'

For every day in the test period

1: Determine for each company which orders are outsourced

2: Compute the inter-depot costs based on the outsourced orders

If necessary, change the inter-depot costs to the new inter-depot costs

Determine the costs after outsourcing

For every day in the test period

3: Compute for each company the fees and inter-depot costs paid, fees received and the routing cost after the outsourcing and the transshipment

The explanation of Algorithm 5.2.2 is as follows. First, each company determines, given the outsource pricing, which orders it will outsource. For this we use the LNS-Faster, which is a faster variant of the LNS-Fast (Chapter 3). The details of the LNS-Faster heuristic can be found in Appendix D. However, the LNS-Faster uses random elements which imply that the solution can be different each execution. Therefore, we run the LNS-Faster three times per company and use the best found solution for each company. How the inter-depot costs are calculated can be found in Section 5.3. If the inter-depot costs deviate to much from the costs that were used, then, the inter-depot costs are changed to the new inter-depot costs. When the inter-depot costs are updated, the companies need to reevaluate which orders they will outsource and the companies use the previous best solution as a starting point for the LNS-Faster. Finally, the companies calculate the costs for delivering the orders after the outsourcing. In other words, the costs for delivering the orders that they did not outsource together with the orders that are outsourced to them. Again, each company runs the LNS-Faster three times and the best solution is used. We assume that the fee consists of a fixed part and a part which depends on the demand of the order. For simplicity, we assume that the fee is the same for each company. The final assumption is that the fee is such that there is no discrimination where the order came from. In other words, $ip = \alpha + \beta \cdot q$ where q is the demand of the order.

5.3 Test instance

For our test instance we assume that we have 9 companies ($n = 9$) which are located in a 3×3 grid as in Figure 5.1. Each company has one of the parts in Figure 5.1 as a home-region. We assume that the inter-depot transportation is executed using a simple hub-and-spoke system with the middle company, company 5, as hub. The following figure illustrates the hub-and-spoke system.

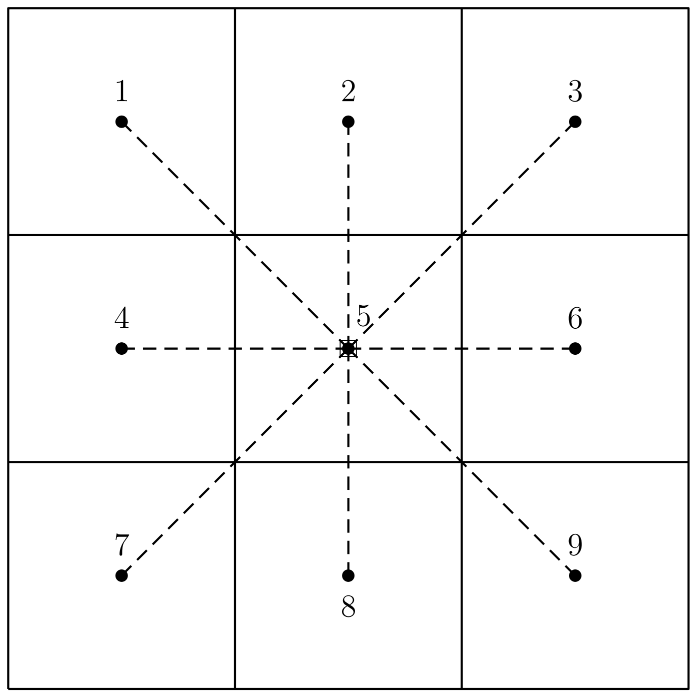

Figure 5.1: The inter-depot transport system.

There are 8 lines in Figure 5.1. Each of these lines depict a route for the inter-depot transport from the depot to the hub and back. The trucks used for the inter-depot transport are longer heavier vehicles with fixed costs 250, variable costs 1.15 per distance driven, and capacity 54 pallets. The additional costs for sorting the order at the hub are 3. The inter-depot costs of an order from company i to company j with demand q is defined as $ID_{ij}(\mathbf{p}(.)) = 3 + (\zeta_i + \zeta_j)q$, where ζ_i (ζ_j) denotes the unit costs of transporting an order from the depot of company i (j) to the hub or the other way around. To be more precise, the number of vehicles required on the line of company $i \in N\backslash\{5\}$ is given by $m_i = \lceil\frac{\max\{In_i, Out_i\}}{Q}\rceil$, where In_i is the demand that goes from the hub to company i and Out_i is the demand that goes to the hub from company i. Then, $\zeta_i = \frac{m_i F_i}{In_i + Out_i}$, where F_i denotes the costs per truck from depot i to the hub and back. For the first iteration, the ζ's are estimated by using the ζ's of the nearest fee. Note that it could be that there is an imbalance in the demand going in and out. One way to solve this imbalance is to have one ζ_i^{Out} for the flow going to the hub and one ζ_i^{In} for the flow going to the company. Then, the ζ of the flow which is greater can be increased, but it should not be more than $\frac{m_i F_i}{\max\{In_i, Out_i\}}$, and the other can be decreased, but it should not be less than 0. For simplicity, we will not amend the costs in case of an imbalance. The following example shows the calculation of ζ_2.

Example 5.3.1 Suppose that there is only one day in the test period. Company 2 outsources orders with a total volume of 146.75 pallets. The other companies outsource orders with a total volume of 78.75 pallets to company 2. The number of routes needed is 3 since $146.75/54 \approx 2.7$. The total distance from the depot of company 2 to the depot of company 5 and back is 197. This implies that the inter-depot costs of this line is $3 \cdot (250 + 1.15 \cdot 197) \approx 1430$. This 1430 needs to be divided over the orders, which is done by usage. In other words, the costs of transporting an order with size q pallets from company 2 to the depot (or the other way around) is $(1430/(146.75 + 75.75)) \cdot q \approx 6.43q$. Note that costs for an order from or to company 2 has the same costs structure regardless the imbalance. Typically, there are more days and the total transportation costs on a line is divided by the total amount of orders to get more robust inter-depot costs for an order. $\triangleleft$

Since we like to represent larger and smaller orders, as well as large and small companies, and urban and rural areas, we have defined our instances as follows. We assume that there is data for ten different days. On each of these days, the companies have orders which need to be delivered. The number of orders that a company has in its region on a given day is a random integer in $[Inside \cdot (1 - dev), Inside \cdot (1 + dev)]$, where dev is a random number in $[-Dev, Dev]$. Similarly, the number of orders outside its region is random integer in $[Outside \cdot (1 - dev), Outside \cdot (1 + dev)]$. Each order outside its region is assigned, with a certain probability, to the region of another company. Similar as in practice, it is more likely that an order has its location close to the depot of the company and it is more likely to be located in an urbanized (industrialized) area. Therefore, we assign a weight to each region which is based on the distance and the type of region. The weight for the region of company

j is given by $\frac{Type_j}{1+d_j}$, where d_j is the distance between the depot of the company who owns the order and the depot of company j, and $Type_j$ reflects whether it is urbanized or not. There are three levels of urbanization, rural which has value 1, normal which has value 2, and urban which has value 4. The location of an order, within a region, is random and the demand of the order is drawn according to the distribution in Table 5.1.

Ordersize	Prob
$\frac{1}{8}$	$\frac{7}{64}$
$\frac{1}{4}$	$\frac{7}{64}$
$\frac{3}{8}$	$\frac{7}{64}$
$\frac{1}{2}$	$\frac{7}{64}$
1	$\frac{5}{16}$
2	$\frac{1}{16}$
3	$\frac{1}{16}$

Ordersize	Prob
4	$\frac{1}{16}$
5	$\frac{1}{96}$
6	$\frac{1}{96}$
7	$\frac{1}{96}$
8	$\frac{1}{96}$
9	$\frac{1}{96}$
10	$\frac{1}{96}$

Table 5.1: The demand distribution of the orders, expressed as number of pallets.

The characteristics of the companies and regions is presented in Table 5.2.

Company/region	Inside	Outside	Dev	Urbanization
1	50	100	5%	Normal
2	100	200	20%	Normal
3	50	100	10%	Rural
4	25	75	10%	Normal
5	50	100	20%	Normal
6	50	150	10%	Urban
7	100	200	10%	Urban
8	75	100	10%	Normal
9	25	50	10%	Rural

Table 5.2: The characteristics of the companies and regions.

Each company has enough trucks available to deliver all its orders including the orders that are outsourced to the company. The trucks for delivering the orders have fixed costs 50, variable costs 1 per distance and a capacity of 25. Finally, we desire a pricing scheme which is robust for multiple days. Hence, we use the first four days to determine the pricing scheme and the remaining six are for verification.

5.4 Test results

Table 5.3 presents the total costs (on the first 4 days) for no cooperation (No coop), which implies that no orders will be outsourced, for all outsourced (All out), which implies that all orders that can be outsourced are outsourced, for no fee (No fee), which implies the decision on which orders to outsource is only based on the inter-depot costs, and finally, for the outsource costs based on the estimated costs (Estimated). For Estimated we estimate with the Huijink(5) estimation (Algorithm 4.2.8) what the costs are for delivering an order. The fee corresponding with these estimated costs is $ip = 2.16+5.10 \cdot q$. In Table 5.3, Costs corresponds to the costs over the test-period, Profit corresponds to the relative gap to no cooperation, Out corresponds to the percentage of orders (averaged over the 4 days) that are outsourced and similarly Cap corresponds to the percentage of demand that is outsourced.

Type	Costs	Profit (%)	Out (%)	Cap(%)
No coop	172,990	-	0	0
All out	159,748	7.66	100	100
No fee	153,884	11.04	85.88	78.24
Estimated	150,728	12.87	69.30	58.64

Table 5.3: The costs of four basic strategies.

The costs for no cooperation are obtained by improving the solution found by the LNS-Fast heuristic (Chapter 3) as follows. The best solution is iteratively perturbed and followed by the LNS-Fast until no improvement was found for a number of iterations. For the other two strategies, all outsourced and no fee, the method in Section 5.2 is used. The results show that incorporating the inter-depot costs leads to less costs than just outsourcing everything or no cooperation. Furthermore, the results show that using the estimated fee leads to the least costs of those four methods.

In Table 5.4, we compare the performance of the Huijink(5) estimation (Chapter 4) to the LNS-Faster. First, we estimate with the Huijink estimation (Algorithm 4.2.8) what the costs are for delivering an order. The fee corresponding with these estimated costs is $ip = 2.16 + 5.10 \cdot q$. The LNS-Faster is used twice in Algorithm 5.2.2, first to determine which orders are outsourced and then to calculate the routing costs. This method is called the LNS-LNS in Table 5.4. Esti-Esti uses the Huijink estimation instead of the LNS-Faster for determining which orders to outsource and uses the estimation instead of the LNS-Faster for calculating the routing costs. Finally, Esti-LNS uses the Huijink estimation for determining which orders to outsource and the LNS-Faster for determining which orders to outsource.

Type	Costs	Profit (%)	Out (%)	Cap (%)
Esti-Esti	142,401	-	35.51	36.02
Esti-LNS	173,258	-0.15	35.51	36.02
LNS-LNS	150,728	12.87	69.30	58.64

Table 5.4: The estimated costs and the true costs when $ip = 2.16 + 5.10 \cdot q$.

Table 5.4 shows that the estimation underestimates the delivery costs which implies that the estimation outsources less orders than when the LNS-Faster decides which orders to outsource. Moreover, the LNS-Faster outsources smaller orders while the estimation has more or less an equal percentage on orders and capacity outsourced.

The previous results show that the LNS-Faster has more reliable results than the estimation method. Hence, we use the LNS-Faster in our Kriging model. Figure 5.2 and Figure 5.3 show the results of the Kriging model, where the dots are the calculated parameter configurations. The costs of the calculated configurations can be found in Table D.3.

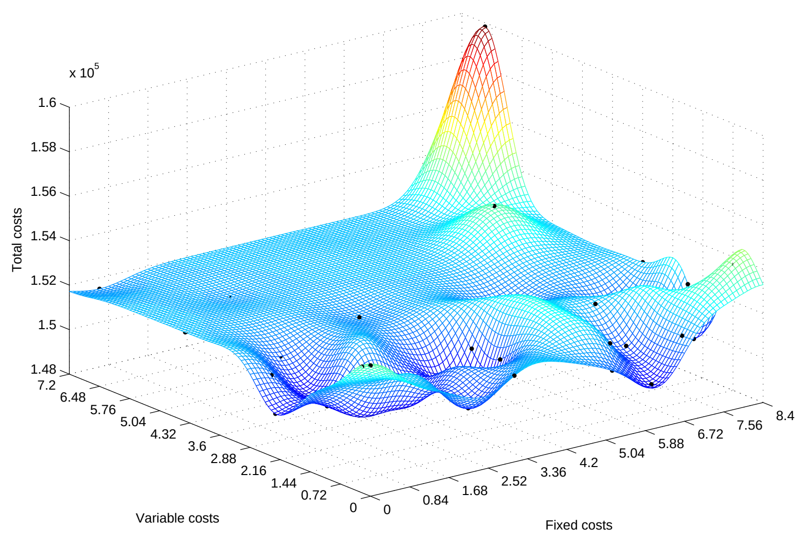

Figure 5.2: The Kriging model.

The best configuration is $ip = 3.86 + 1.95 \cdot q$. There are four calculated configurations around the best configuration and they all have costs below $149,500$. The

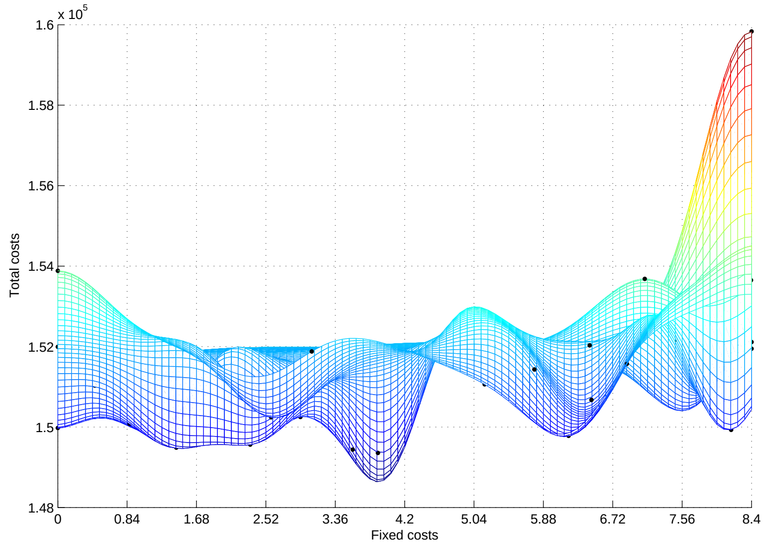

Figure 5.3: The Kriging model alternative view.

results of this configuration are presented in Table 5.5, where Costs corresponds to the costs of the period, CostsDay corresponds to the average costs per day in the period, ID-costs corresponds to the average of the true inter-depot transport costs per day (without the fee of 3 per order), and the ID-Paid corresponds to the average that the companies paid for the inter-depot transport per day, again without the fee of 3 per order.

Type	Costs	CostsDay	Profit (%)	Out(%)	Cap (%)	ID-Costs	ID-Paid
Test period	148,688	37,172	14.05	72.48	64.55	11,542	11,542
Operational period	225,899	37,650	12.11	70.29	62.94	11,454	11,585
Total	374,587	37,459	12.16	71.17	63.58	11,489	11,567

Table 5.5: The results of the best point $ip = 3.86 + 1.95 \cdot q$.

The test period consists of the first four days and the operational period of the last six days. This test period is used to determine the fee and inter-depot costs. For the operational period, we use the fee and inter-depot costs (the ζ's) obtained in the test period. From Table 5.5, we see that the companies pay on the operational days slightly too much for the inter-depot costs compared to the true average inter-depot costs. The best fee is $ip = 3.86+1.95\cdot q$ while the estimated fee is $ip = 2.16+5.10\cdot q$. Due to the much lower variable part (1.95 instead of 5.10), the companies outsource more demand than with the estimated fee. Each company has a profit on all of the first 4 days. However, there are several companies with losses on one of the last six days. Companies can have a loss when delivering the orders outsourced to them costs more than the fee they receive. Furthermore, it could imply that the first four days are not representative enough. On average, each company makes a profit, which is shown in Table 5.6. The average profit ranges from 6.5% up to 22%. The companies with the most profit are company 5, which has much less inter-depot costs compared to the other companies, and company 9, which is the smallest company. The higher profit in the test period can (partly) be explained by the fact that the parameters are tuned on the test period and not on the operational period.

Company	Test period	Operational period	Average
1	15.99	14.00	14.81
2	7.29	5.99	6.51
3	14.18	13.36	13.69
4	16.62	8.61	11.82
5	24.45	20.70	22.18
6	12.91	10.98	11.77
7	12.89	8.98	10.61
8	12.46	12.65	12.57
9	20.19	20.91	20.64

Table 5.6: The average profit of the companies in (%).

Table 5.7 presents the characteristics of each company averaged over the ten days, where NoCoop corresponds to the costs without cooperation, VRP corresponds to the costs of the routing after the transshipment of the orders, Paid corresponds to the outsource costs paid, which consists of the fee paid to the other companies and the inter-depot costs including the sorting costs. Rec corresponds to the fee that the company receives, NOut corresponds to the number of orders outsourced, NIn to the number of orders that are outsourced to the company, CapOut to the demand outsourced, CapIn to the demand that is outsourced to the company, and ID-Costs corresponds to the true inter-depot costs (without the fee of 3 per order) on the line of the company.

Company 1 has on average an almost perfect balance between outsourced demand and the demand that the company receives. This is, however, deceiving since company 1 outsources a demand of 68.13 while it receives 135.25 on day three. The remaining results are in line with the characteristics of the company.

The method in Section 5.2 uses heuristics. This can imply that companies do not make the optimal decisions or that the routing after the outsourcing can be improved. Moreover, the deep valley in Figures 5.2 and 5.3 indicate that the pricing

	NoCoop	VRP	Paid	Rec	NOut	NIn	CapOut	CapIn	ID-Costs
1	4358.92	2321.83	1850.89	459.46	72.20	73.30	92.93	91.04	1368.94
2	7225.53	3987.16	3241.28	473.32	123.60	73.50	165.36	93.68	1667.93
3	4334.77	2236.36	1739.16	234.13	65.80	36.00	78.99	49.39	1140.78
4	3120.70	1994.45	1216.60	459.28	48.60	74.20	61.51	89.88	1000.76
5	3835.75	2068.11	1688.23	771.28	89.60	117.80	124.70	161.14	-
6	5659.06	2951.84	2878.92	837.85	115.30	129.60	160.56	174.29	1858.55
7	7536.84	4241.97	3359.11	864.14	129.40	128.90	178.84	187.78	2395.64
8	4340.71	2557.77	1706.91	469.79	69.20	74.70	84.83	91.01	1143.72
9	2603.24	1359.09	1002.69	295.95	36.70	42.40	44.08	53.60	912.62

Table 5.7: The average characteristics per day, over the ten days.

mechanism is likely not robust. This point is strengthened by the fact that the profits are much less in the operational period. The deep valley could also imply that the randomness of the heuristic is still an issue. The regressing Kriging model (Forrester et al. (2008)), which takes this randomness into account by allowing for an error around each calculated configuration, is shown in Figures 5.4 and 5.5.

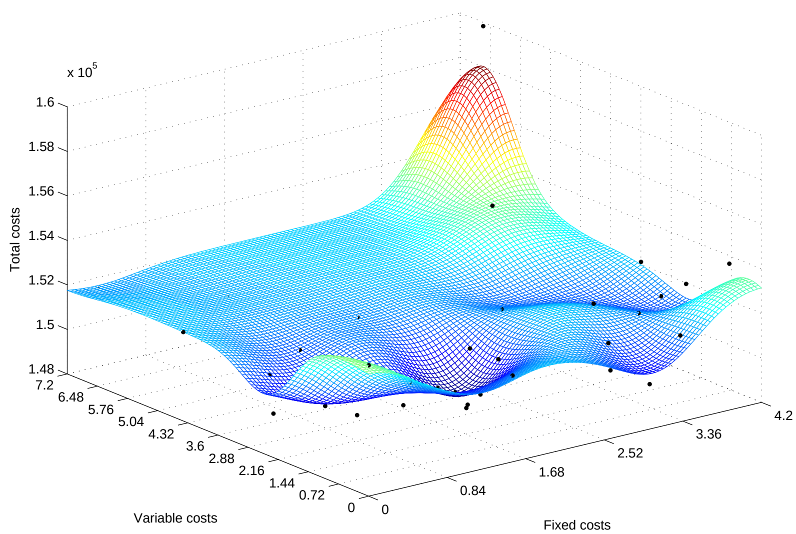

Figure 5.4: The regressing Kriging model.

The difference between the two Kriging models is that the regressing model is smoother. The regressing model has the same interesting area, but it assumes that the calculated configurations in the interesting area are too optimistic.

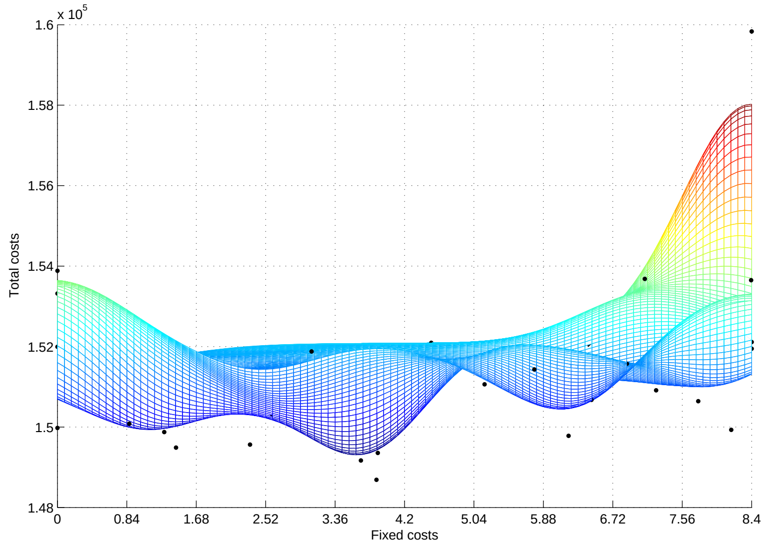

Figure 5.5: The regressing Kriging model alternate view.

5.5 Conclusions and recommendations

We analyze the pricing based structure and introduce a new test-instance. Furthermore, we present a solution method for this complex problem which is based on Kriging and the heuristics of Chapter 3. The results show that the estimations can not be used on this instance due to both a too large gap and a too large difference in the outsourced orders. Furthermore, the results show that Kriging is able to determine promising areas of parameter configurations when there are not too many parameters. However, there is some variance in the outcomes since our solution method uses heuristics which contain randomness. When verifying the pricing, companies have a negative result on certain days. It could be that something which is good for the coalition as a whole is bad for one company. However, it is something which has to be taken into account when the coalition implements the method. The differences between the results on the first four and the last six raise the question whether the first four instances are representative. Another reason for the negative results in a few instances could be due to the fact that the companies are different and, therefore, each company requires a different fee. Possible fields of future research are to reduce the variance in the results, change the assumption on the inter-depot costs, for example, adjust the costs as proposed or to incorporate parameters for the inter-depot costs, and use other surrogate models than Kriging.

Furthermore, a different objective can be used as well as a company dependent fee.

Chapter 6

Transferable utility games, nucleoni and bankruptcy

6.1 Cooperative considerations in transportation

The way to allocate jointly generated profits is important for success when transportation companies cooperate using the central planning structure. Many central planning structures for transportation companies have failed on mistrust on the applied profit allocation scheme (Cruijssen et al. (2007)). To prevent this mistrust, several types of allocation schemes are investigated in the literature on cooperation between transportation companies. The first class of schemes are corporate rules. Corporate rules use a specific characteristic of the underlying transportation problem, for example proportional to volume or proportional to the stand alone transportation costs (Cruijssen et al. (2007)). A more complete list of corporate rules as compensation schemes can be found in Table 6.1. For an explanation of these rules we refer to their respective papers.

However, using a single characteristic, it could be the case that the true contribution of a company to the optimal cooperation is neglected. For example, using the volume will give a company with many orders a large share of the savings, even though it could well be that this company did not generate any savings for the cooperation. Compensation schemes which take the marginal costs or savings of a company into account are typically solution concepts of cooperative game theory. These solution concepts in general use the worth of coalitions, potentially an operating subgroup, to determine an appropriate allocation. This worth of the coalition estimates the minimal costs for delivering all the orders of the companies belonging to the coalition in a consistent way. The game theoretic solution concepts used in

Corporate rule	*Papers*
Equal split of the gains	Fleischmann (1999)
Equal unit costs	Fleischmann (1999)
Proportional to volume	Fleischmann (1999); Cruijssen et al. (2007); Frisk et al. (2010)
Proportional to the number of orders delivered	Cruijssen et al. (2007)
Proportional to the stand alone transportation costs	Fleischmann (1999); Cruijssen et al. (2007); Lui et al. (2010)
Proportional to the distance traveled	Cruijssen et al. (2007)
Proportional to the number of orders	Cruijssen et al. (2007)
Shadow Prices	Frisk et al. (2010)

Table 6.1: Corporate rules for central planning structures for transportation.

the literature on cooperation between transportation companies can be found in Table 6.2. For an explanation of these concepts we refer to their respective papers.

Game theoretic rule	*Papers*
Equal Charge method	Frisk et al. (2010)
Alternative Cost Avoided Method	Frisk et al. (2010); Hezarkhani et al. (2015)
Cost Gap Method	Frisk et al. (2010)
Shapley value	Frisk et al. (2010); Lui et al. (2010); Krajewska et al. (2008); Hezarkhani et al. (2015)
Core guaranteed Shapley mechanism	Dai and Chen (2012)
nucleolus	Frisk et al. (2010); Lui et al. (2010); Hezarkhani et al. (2015)
Equal Profit Method	Frisk et al. (2010); Audy et al. (2011); Hezarkhani et al. (2015)
Modified Equal Profit Method	Audy et al. (2011)
Weighted Relative Savings Model	Lui et al. (2010)
Core guaranteed proportional mechanism	Dai and Chen (2012)
Core guaranteed contribution-based mechanism	Dai and Chen (2012)
Proportional	Hezarkhani et al. (2015)
Modified proportional	Hezarkhani et al. (2015)

Table 6.2: Game theoretic solution concepts used in central planning structures for transportation.

In the remainder of this dissertation, we consider a pure game theoretic setting.

In other words, we assume that each coalition has a worth and we investigate fair allocations. More specifically, we focus on the per capita nucleolus (Grotte (1970)) and the proportionate nucleolus. Both these allocations are variants of the nucleolus (Schmeidler (1969)). The nucleolus (Schmeidler (1969)) lexicographically minimizes the maximal dissatisfaction of coalitions, where the dissatisfaction of a coalition for a given allocation is expressed as the difference between the worth of the coalition and the joint payoff that the coalition receives according to this allocation, while the per capita nucleolus (Schmeidler (1969)) lexicographically minimizes the maximal dissatisfaction per player of coalitions. The main difference between the nucleolus and the per capita nucleolus is that the per capita nucleolus incorporates the size of the coalitions. An important result for both the nucleolus and the per capita nucleolus is that the resulting allocation is an element of the core (Gillies (1953)), if the core is non-empty. The remainder of this chapter is as follows. In Section 6.2, we provide the basic notions of cooperative game theory. Section 6.3 provides an overview of the three Kohlberg (1971) type of characterizations of the nucleolus. These characterizations for the nucleolus are also valid for the per capita nucleolus. Finally, bankruptcy problems, which deals with the question how to divide the remaining estate over the claimants, are introduced in Section 6.4.

6.2 Transferable utility games and nucleoni

A *transferable utility* TU-game is defined by a pair (N, v), where $N = \{1, ..., n\}$ is the finite set of players and $v : 2^N \to \mathbb{R}$ is the *characteristic function*. The set of all TU-games with player set N is denoted by TU^N and a TU-game with player set N is abbreviated by v. For every coalition $S \in 2^N$, $v(S)$ is called the *worth* of the coalition, which is the maximal joint amount of money that the coalition can obtain on its own by cooperating in an optimal way, with $v(\varnothing) = 0$ by convention.

The cardinality of a coalition $S \in 2^N$ is denoted by $|S|$. By $\mathbb{R}^N$, we denote the set of all real-valued vectors with $|N|$ elements in which each coordinate corresponds to a player $i \in N$. For $S \in 2^N$, we denote by $e^S \in \mathbb{R}^N$ the vector for which $e^S_i = 1$ for all $i \in S$ and $e^S_i = 0$ for all $i \in N \backslash S$.

An *efficient* vector is a vector $x \in \mathbb{R}^N$ such that $\sum_{i \in N} x_i = v(N)$. The *imputation* *set*, $I(v)$, is defined by

$$I(v) = \{x \in \mathbb{R}^N \mid x_i \geq v(\{i\}) \text{ for all } i \in N \text{ and } \sum_{i \in N} x_i = v(N)\}.$$

The core, $Core(v)$ (Gillies (1953)), consists of all imputations for which no coalition would be better off if it separated itself from the grand coalition. Formally, the core is defined by

$$Core(v) = \{x \in I(v) \mid \sum_{i \in S} x_i \geq v(S) \text{ for all } S \in 2^N\}.$$

For $x, y \in \mathbb{R}^t$ we have $x \leq_L y$, *i.e.*, x is *lexicographically smaller* than (or equal to) y, if $x = y$ or if there exists an $\ell \in \{1, ..., t\}$ such that $x_k = y_k$ for all $k \in \{1, ..., \ell - 1\}$ and $x_\ell < y_\ell$.

Let $v \in TU^N$. The *excess* $exc(v, S, x)$ of coalition $S \in 2^N$ for an imputation $x \in I(v)$ is defined by

$$exc(v, S, x) = v(S) - \sum_{i \in S} x_i.$$

For a game $v \in TU^N$ and imputation $x \in I(v)$, the *excess vector* $\theta(x) \in \mathbb{R}^{2^{|N|}}$ has as its coordinates the excesses of all $2^{|N|}$ coalitions arranged in a weakly decreasing order, *i.e.*, $\theta_k(x) \geq \theta_{k+1}(x)$ for all $k \in \{1, ..., 2^{|N|} - 1\}$. The nucleolus is defined as follows.

Definition 6.2.1 *(cf. Schmeidler (1969))*
Let $v \in TU^N$ be such that $I(v) \neq \varnothing$. The nucleolus, *$n(v)$, is the unique imputation such that $\theta(n(v)) \leq_L \theta(y)$ for all $y \in I(v)$.*

For a game $v \in TU^N$ and an imputation $x \in I(v)$ we define the *per capita excess* of any non-empty coalition $S \in 2^N \backslash \{\varnothing\}$ by

$$exc^P(v, S, x) = \frac{v(S) - \sum_{i \in S} x_i}{|S|}.$$

The *per capita excess vector* $\theta^P(x) \in \mathbb{R}^{2^{|N|}-1}$ has as its coordinates the per capita excesses of all non-empty coalitions arranged in a weakly decreasing order, in other words, $\theta^P_k(x) \geq \theta^P_{k+1}(x)$ for all $k \in \{1, ..., 2^{|N|} - 2\}$.

Definition 6.2.2 *(cf. Grotte (1970))*
Let $v \in TU^N$ be such that $I(v) \neq \varnothing$. Then, the per capita nucleolus*, $pcn(v)$, is the unique imputation such that $\theta^P(pcn(v)) \leq_L \theta^P(y)$ for all $y \in I(v)$.*

6.3 Characterizations of nucleoni using balanced collections

Besides the original definition of the nucleolus, there exist multiple characterizations (cf. Kohlberg (1971); Groote Schaarsberg et al. (2013)) that use balanced collections. One of the advantages of these characterizations is that they provide ways to quickly determine whether an imputation is the nucleolus or not. We use the following variant, which uses the following definitions: A map $\rho : 2^N \backslash \{\varnothing\} \to [0, \infty)$ is called *balanced* if

$$\sum_{S \in 2^N \backslash \{\varnothing\}} \rho(S) e^S = e^N.$$

Furthermore, a collection $\mathcal{B} \subset 2^N \backslash \{\varnothing\}$ of coalitions is called *balanced* if there exists a balanced map ρ on N such that

$$\mathcal{B} = \{S \in 2^N \backslash \{\varnothing\} \mid \rho(S) > 0\}.$$

We call the grand coalition N and the empty coalition $\varnothing$ trivial. Let $x \in I(v)$ and define $\mathcal{B}_1(v, x)$ to be the set of the non-trivial coalitions for which the excess with imputation x is the highest. Formally,

$$\mathcal{B}_1(v, x) = \Big\{S \in 2^N \backslash \{\varnothing, N\} \mid exc(v, S, x) \geq exc(v, T, x) \text{ for all } T \in 2^N \backslash \{\varnothing, N\}\Big\}.$$

Recursively, for $k = 2, 3, ...$ the sets $\mathcal{B}_k(v, x)$ are defined by,

$$\mathcal{B}_k(v, x) = \Big\{S \in 2^N \backslash \Big(\{\varnothing, N\} \cup \bigcup_{\ell=1}^{k-1} \mathcal{B}_\ell(v, x)\Big) \mid exc(v, S, x) \geq exc(v, T, x) \\ \text{for all } T \in 2^N \backslash \Big(\{\varnothing, N\} \cup \bigcup_{\ell=1}^{k-1} \mathcal{B}_\ell(v, x)\Big)\Big\}.$$

Since $2^N \backslash \{\varnothing, N\}$ is finite, it is clear that there exists a unique $t(v, x) \in \mathbb{N}$, such that

$$\begin{cases} \mathcal{B}_k(v, x) \neq \varnothing \text{ for all } k \in \{1, ..., t(v, x)\} \\ \mathcal{B}_k(v, x) = \varnothing \text{ for all } k \in \{t(v, x) + 1, ...\} \end{cases}$$

Proposition 6.3.1 *(cf. Kohlberg (1971))*
Let $v \in TU^N$ be such that $Core(v) \neq \varnothing$ and let $x \in I(v)$. Then, $x = n(v)$ if and only if $\bigcup_{\ell=1}^{k} \mathcal{B}_\ell(v,x)$ is balanced for all $k \in \{1, ..., t(v,x)\}$.

An alternative characterization is provided by Groote Schaarsberg et al. (2013). Let $\mathcal{D} \subseteq 2^N$ and let $H(\mathcal{D})$ be as follows:

$$H(\mathcal{D}) = \Big\{ S \in 2^N \mid e^S \in span(e^N, \{e^T\}_{T\in\mathcal{D}}) \Big\},$$

where *span* denotes the linear hull. Formally,
$span(e^N, \{e^T\}_{T\in\mathcal{D}}) = \Big\{ \sum_{S\in\mathcal{D}\cup\{N\}} \gamma_S e^S \mid \gamma_S \in \mathbb{R} \text{ for all } S \in \mathcal{D}\cup\{N\} \Big\}$ and note that $H(\{\varnothing\}) = \{\varnothing, N\}$.

Proposition 6.3.2 *(cf. Groote Schaarsberg et al. (2013))*
Let $v \in TU^N$ be such that $Core(v) \neq \varnothing$ and let $x \in I(v)$. Then, $x = n(v)$ if and only if there exists a sequence $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_\tau$ of non-empty subcollections of $2^N \backslash \{\varnothing, N\}$ with the following properties:

(i) for all $r \in \{1, ..., \tau\}$ the collection $\bar{\mathcal{D}}_r = \bigcup_{k=1}^{r} \mathcal{D}_k$ is balanced.

(ii) there exists a sequence of real numbers $\gamma_1, \gamma_2, ..., \gamma_\tau$ such that $exc(v,T,x) = \gamma_r$ for every $T \in \mathcal{D}_r$ and all $r \in \{1, ..., \tau\}$ and that $\gamma_1 \geq \gamma_2 \geq ... \geq \gamma_\tau$.

(iii) for all $T \in 2^N \backslash \{\{\varnothing, N\} \cup \bar{\mathcal{D}}_\tau\}$ it holds that $T \in H(\{S \in \bar{\mathcal{D}}_\tau : exc(v,S,x) \geq exc(v,T,x)\})$.

Example 6.3.1 Consider the four person game $v \in TU^N$ with $N = \{1,2,3,4\}$ given by:

S	$\{1\}$	$\{2\}$	$\{3\}$	$\{4\}$	$\{1,2\}$	$\{1,3\}$	$\{1,4\}$	$\{2,3\}$
$v(S)$	0	0	0	0	11	0	0	0

S	$\{2,4\}$	$\{3,4\}$	$\{1,2,3\}$	$\{1,2,4\}$	$\{1,3,4\}$	$\{2,3,4\}$	$\{1,2,3,4\}$
$v(S)$	0	60	30	30	80	90	100

Then, $n(v) = (5, 13, 41, 41)$ and the sorted excesses of the non-trivial coalitions are as follows:

S	$\{1\}$	$\{2,3,4\}$	$\{1,2\}$	$\{1,3,4\}$	$\{2\}$	$\{3,4\}$	$\{1,2,3\}$	$\{1,2,4\}$
$exc(v, S, n(v))$	-5	-5	-7	-7	-13	-22	-29	-29

S	$\{3\}$	$\{4\}$	$\{1,3\}$	$\{1,4\}$	$\{2,3\}$	$\{2,4\}$
$exc(v, S, n(v))$	-41	-41	-46	-46	-54	-54

These excesses imply that $\mathcal{B}_1(v, n(v)) = \{\{1\}, \{2,3,4\}\}$, $\mathcal{B}_2(v, n(v)) = \{\{1,2\}, \{1,3,4\}\}$, $\mathcal{B}_3(v, n(v)) = \{\{2\}\}$, $\mathcal{B}_4(v, n(v)) = \{\{3,4\}\}$, $\mathcal{B}_5(v, n(v)) = \{\{1,2,3\}, \{1,2,4\}\}$, $\mathcal{B}_6(v, n(v)) = \{\{3\}, \{4\}\}$, $\mathcal{B}_7(v, n(v)) = \{\{1,3\}, \{1,4\}\}$, $\mathcal{B}_8(v, n(v)) = \{\{2,3\}, \{2,4\}\}$, and $t(v, n(v)) = 8$.
Note that $H\Big(\bigcup_{\ell=1}^{5} \mathcal{B}_\ell(v, n(v))\Big) = 2^N$ and that $\mathcal{B}_4(v, n(v)) \subset H\Big(\bigcup_{\ell=1}^{2} \mathcal{B}_\ell(v, n(v))\Big)$. Hence, regarding the characterisation in Proposition 6.3.2, one can conclude that $\tau = 4$, $\mathcal{D}_1 = \{\{1\}, \{2,3,4\}\}$, $\mathcal{D}_2 = \{\{1,2\}, \{1,3,4\}\}$, $\mathcal{D}_3 = \{\{2\}\}$, and $\mathcal{D}_4 = \{\{1,2,3\}, \{1,2,4\}\}$. $\triangleleft$

Both Proposition 6.3.1 and 6.3.2 require that (a part of) the coalitions are put into a sequence of collections, and that all coalitions in a collection have the same excess. Moreover, both sequences of collections have to satisfy the same balancedness requirement. However, there are two important differences between the propositions. First, in Proposition 6.3.2 it is allowed that several collections have the same excess as opposed to Proposition 6.3.1. In other words, in Proposition 6.3.2 it is allowed to split a large Kohlberg collection into multiple smaller collections. Second, Proposition 6.3.2 states that a non-trivial coalition either belongs to a collection or it is in the span of the collections with higher excesses, while Proposition 6.3.1 states that each non-trivial coalition belongs to a collection. Hence, it is possible to use a subset of the Kohlberg collections to determine whether an imputation is the nucleolus or not, in other words, some of the Kohlberg collections are irrelevant. Finally, note that the sequence of the original Kohlberg collections satisfies the three properties of Proposition 6.3.2.

We formulate another variant which exploits the idea of Groote Schaarsberg (Groote Schaarsberg et al. (2013)) that not all Kohlberg collections are relevant. For collections $\mathcal{D} \subseteq 2^N$, denote by $\mathcal{F}(\mathcal{D})$ the set of the *free* coalitions. In other words, coalitions which are not in the span of $\mathcal{D}$. Formally, the set of free coalitions is given by

$$\mathcal{F}(\mathcal{D}) = 2^N \backslash H(\mathcal{D}).$$

Proposition 6.3.3 *Let $v \in TU^N$ be such that $Core(v) \neq \varnothing$ and let $x \in I(v)$. Then, $x = n(v)$ if and only if there exists a sequence $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_\tau$ of non-empty subcollections of $2^N \backslash \{\varnothing, N\}$ such that for $\bar{\mathcal{D}}_{r-1} = \bigcup_{\ell=1}^{r-1} \mathcal{D}_\ell$, $\mathcal{D}_r \subseteq \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ for all $r \in \{1, ..., \tau\}$, that satisfies the following properties:*

(A) *for all $r \in \{1, ..., \tau\}$ the collection $\bar{\mathcal{D}}_r = \bigcup_{k=1}^{r} \mathcal{D}_k$ is balanced and $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$.*

(B) *for all $r \in \{1, ..., \tau\}$ and all $T \in \mathcal{D}_r$ it holds that*
$$exc(v, T, x) = \max_{S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})} exc(v, S, x).$$

Proof: "only if part". We show how to define a sequence $\mathcal{D}_1, ..., \mathcal{D}_\tau$ of relevant collections from the Kohlberg collections. In this proof, we abbreviate $\mathcal{B}_k(v, n(v))$ by $\mathcal{B}_k$.

Let $\mathcal{B}_1, ..., \mathcal{B}_{t(v,n(v))}$ be the sequence of Kohlberg collections of the nucleolus. Then, determine the sequence of relevant collections with the following algorithm:

1: $r = 1$ and $\mathcal{D}_1 = \mathcal{B}_1$
2: **while** $\mathcal{F}(\bar{\mathcal{D}}_r) \neq \varnothing$ **do**
3: $\quad r = r + 1$
4: $\quad k_r = \min\left\{\ell \in \{1, ..., t(v, n(v))\} \mid \mathcal{B}_\ell \cap \mathcal{F}(\bar{\mathcal{D}}_{r-1}) \neq \varnothing\right\}$
5: $\quad \mathcal{D}_r = \mathcal{B}_{k_r} \cap \mathcal{F}(\bar{\mathcal{D}}_{r-1})$
6: **end while**
7: $\tau = r$

By construction we have that $\mathcal{D}_r \subseteq \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ for all $r \in \{1, ..., \tau\}$ and that $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$. Left to show is that the sequence satisfies (B) and the remaining part of (A).

For each $r \leq \tau$, we have $\mathcal{D}_r = \mathcal{B}_{k_r} \cap \mathcal{F}(\bar{\mathcal{D}}_{r-1})$, which implies that the coalitions in collection $\mathcal{D}_r$ have maximum excess with respect to the nucleolus over the set $\mathcal{F}(\bar{\mathcal{D}}_{r-1})$. This gives (B).

Left to prove is the balancedness of the collections $\bar{\mathcal{D}}_r$ for all $r \in \{1, ...\tau\}$, which is shown by induction.
Basis: $\bar{\mathcal{D}}_1$ is balanced, being equal to $\mathcal{B}_1$. Let $r \in \{2, \ldots, \tau\}$ and assume that $\bar{\mathcal{D}}_{r-1}$ is balanced. Define $\bar{\mathcal{B}}_{k_r} = \bigcup_{\ell=1}^{k_r} \mathcal{B}_\ell$ and denote $\mathcal{G} = \bar{\mathcal{B}}_{k_r} \cap H(\bar{\mathcal{D}}_{r-1})$. Then, $\bar{\mathcal{B}}_{k_r}$ is the disjoint union of $\mathcal{G}$ and $\mathcal{D}_r$, *i.e.*, $\mathcal{G} \cap \mathcal{D}_r = \varnothing$ and $\bar{\mathcal{B}}_{k_r} = \mathcal{G} \cup \mathcal{D}_r$. Because $\bar{\mathcal{B}}_{k_r}$ and $\bar{\mathcal{D}}_{r-1}$ are balanced, there exist for both collections a balanced map, *i.e.*, there exists a ρ, where $\rho(T) > 0$ for all $T \in \bar{\mathcal{B}}_{k_r}$, and an α, where $\alpha(T) > 0$ for all $T \in \bar{\mathcal{D}}_{r-1}$ with

$$e^N = \sum_{T \in \bar{\mathcal{B}}_{k_r}} \rho(T) e^T = \sum_{T \in \bar{\mathcal{D}}_{r-1}} \alpha(T) e^T.$$

Furthermore, since $\mathcal{G} \subseteq H(\bar{\mathcal{D}}_{r-1})$, we have that for every $S \in \mathcal{G}$, there exists a vector $\gamma_T^S \in \mathbb{R}^{\bar{\mathcal{D}}_{r-1} \cup \{N\}}$ such that

$$e^S = \sum_{T \in \bar{\mathcal{D}}_{r-1} \cup \{N\}} \gamma_T^S e^T.$$

Denote $\beta_T = \sum_{S \in \mathcal{G}} \rho(S) \gamma_T^S$ for all $T \in \bar{\mathcal{D}}_{r-1} \cup \{N\}$. By substituting the equation above in the balancing equation of $\bar{\mathcal{B}}_{k_r}$, we find

$$\begin{aligned} e^N &= \sum_{T \in \bar{\mathcal{B}}_{k_r}} \rho(T) e^T \\ &= \sum_{S \in \mathcal{G}} \rho(S) e^S + \sum_{T \in \mathcal{D}_r} \rho(T) e^T \\ &= \sum_{S \in \mathcal{G}} \rho(S) \sum_{T \in \bar{\mathcal{D}}_{r-1} \cup \{N\}} \gamma_T^S e^T + \sum_{T \in \mathcal{D}_r} \rho(T) e^T \\ &= \sum_{T \in \bar{\mathcal{D}}_{r-1} \cup \{N\}} \beta_T e^T + \sum_{T \in \mathcal{D}_r} \rho(T) e^T. \end{aligned}$$

Let $\varepsilon \in (0, 1)$ and take a convex combination of the equation above and the balancing equation of $\bar{\mathcal{D}}_{r-1}$:

$$e^N = \varepsilon \Big(\sum_{T \in \bar{\mathcal{D}}_{r-1} \cup \{N\}} \beta_T e^T + \sum_{T \in \mathcal{D}_r} \rho(T) e^T \Big) + (1 - \varepsilon) \Big(\sum_{T \in \bar{\mathcal{D}}_{r-1}} \alpha(T) e^T \Big),$$

which rewrites to

$$e^N = \frac{1}{1 - \varepsilon \beta_N} \Big(\sum_{T \in \bar{\mathcal{D}}_{r-1}} \Big(\varepsilon \beta_T + (1 - \varepsilon) \alpha(T) \Big) e^T + \sum_{T \in \mathcal{D}_r} \rho(T) e^T \Big).$$

Note that $\alpha(T) > 0$ for all $T \in \bar{\mathcal{D}}_{r-1}$ and that $\beta_T \in \mathbb{R}$ for all $T \in \bar{\mathcal{D}}_{r-1} \cup \{N\}$. For ε sufficiently close to 0, this provides a balancing equation for $\bar{\mathcal{D}}_r$. Hence, $\bar{\mathcal{D}}_r$ is balanced, which proves (A).
"if part". Consider a sequence $\mathcal{D}_1, ..., \mathcal{D}_\tau$ of non-empty coalitions such that $\mathcal{D}_r \subseteq \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ for all $r \in \{1, ..., \tau\}$ and that satisfies (A) and (B) of Proposition 6.3.3. We show that this sequence satisfies conditions $(i), (ii)$ and (iii) of Proposition 6.3.2.

Condition (A) implies (i). Furthermore, (B) implies (ii) since $\mathcal{F}(\bar{\mathcal{D}}_r) \subsetneq \mathcal{F}(\bar{\mathcal{D}}_{r-1})$. Property (iii) is inferred as follows. Let $T \in 2^N \backslash \{\{\varnothing, N\} \cup \bar{\mathcal{D}}_\tau\}$. By (A) we have that $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$, hence there exists a unique $r \in \{1, ..., \tau\}$ with $T \in \mathcal{F}(\bar{\mathcal{D}}_{r-1}) \backslash \mathcal{F}(\bar{\mathcal{D}}_r)$. Furthermore, $T \in \mathcal{F}(\bar{\mathcal{D}}_{r-1}) \backslash \mathcal{F}(\bar{\mathcal{D}}_r)$ implies $exc(v, T, x) \leq \gamma_r$ and $exc(v, S, x) \geq \gamma_r$ for all $S \in \bar{\mathcal{D}}_r$. Therefore, we have

$$\begin{aligned} T \in \mathcal{F}(\bar{\mathcal{D}}_{r-1}) \backslash \mathcal{F}(\bar{\mathcal{D}}_r) &= H(\bar{\mathcal{D}}_r) \backslash H(\bar{\mathcal{D}}_{r-1}) \\ &\subseteq H(\bar{\mathcal{D}}_r) \\ &\subseteq H(\{S \in \bar{\mathcal{D}}_\tau : \ exc(v, S, x) \geq \gamma_r\}) \\ &\subseteq H(\{S \in \bar{\mathcal{D}}_\tau : \ exc(v, S, x) \geq exc(v, T, x)\}). \end{aligned}$$

This proves (iii) of Proposition 6.3.2. Hence, Proposition 6.3.3 implies Proposition 6.3.2. □

Example 6.3.2 (Example 6.3.1 continued.) Consider the four person game $v \in TU^N$ with $N = \{1, 2, 3, 4\}$ analyzed in Example 6.3.1 with $n(v) = (5, 13, 41, 41)$. Regarding the characterization in Proposition 6.3.3 one can choose $\tau = 3$, $\mathcal{D}_1 = \{\{1\}, \{2, 3, 4\}\}$, $\mathcal{D}_2 = \{\{1, 2\}, \{1, 3, 4\}\}$, and $\mathcal{D}_3 = \{\{1, 2, 3\}, \{1, 2, 4\}\}$. Note that the four collections used in Example 6.3.1 to illustrate Proposition 6.3.2 do not satisfy the conditions of Proposition 6.3.3. ◁

Wallmeier (1983) showed that the characterization of Kohlberg (1971) can be reformulated to provide a characterization of the per capita nucleolus. Similarly, the characterizations above can be reformulated. In Chapter 6, the following per capita variant of Proposition 6.3.3 is used.

Proposition 6.3.4 *Let $v \in TU^N$ be such that $Core(v) \neq \varnothing$ and let $x \in I(v)$. Then, $x = pcn(v)$ if and only if there exists a sequence $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_\tau$ of non-empty subcollections of $2^N \backslash \{\varnothing, N\}$ such that $\mathcal{D}_r \subseteq \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ for all $r \in \{1, ..., \tau\}$ that satisfy the following properties:*

(A) *for all $r \in \{1, ..., \tau\}$ the collection $\bar{\mathcal{D}}_r = \bigcup_{k=1}^r \mathcal{D}_k$ is balanced and $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$.*

(B) *for all $r \in \{1, ..., \tau\}$ and all $T \in \mathcal{D}_r$ it holds that*
$exc^P(v, T, x) = \max_{S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})} exc^P(v, S, x)$.

6.4 Bankruptcy problems, bankruptcy rules and bankruptcy games

In a bankruptcy problem an insufficient monetary estate has to be divided among a number of claimants, each having a justified claim on this estate. Bankruptcy rules propose general principles and procedures to solve any bankruptcy problem. From the wide variety of bankruptcy rules we mention the constrained equal award rule, the constrained equal loss rule and the Aumann-Maschler rule (cf. Aumann and Maschler (1985)). An overview of bankruptcy rules and their properties can be found in Thomson (2003).

O'Neill (1982) associates a cooperative bankruptcy game with transferable utility to each bankruptcy problem. As a result, game theoretic solution concepts such as the nucleolus (Schmeidler (1969)) can be viewed as bankruptcy rules, too when they are applied to the bankruptcy game associated with the bankruptcy problem. Interestingly, it turns out that the Aumann-Maschler rule coincides with the nucleolus of the corresponding bankruptcy game (Aumann and Maschler (1985)).

A *bankruptcy problem*, for short, a *problem*, is denoted by (N, E, c), where $N = \{1, ..., n\}$ is the set of *claimants*, which will be called *players*, $E \in \mathbb{R}_+$ is the monetary estate that has to be divided over the players, and $c \in \mathbb{R}_+^N$ is the vector of *claims*. By the nature of a problem, the sum of claims exceeds the estate, *i.e.*, $E \leq \sum_{i \in N} c_i$. The class of problems on N is denoted by BR^N and a problem with player set N is abbreviated by (E, c).

A *bankruptcy rule*, which is abbreviated by a *rule*, $f : BR^N \to \mathbb{R}^N$ is a function that assigns to each problem $(E, c) \in BR^N$ a vector $f(E, c) \in \mathbb{R}^N$ such that $\sum_{i \in N} f_i(E, c) = E$ and $0 \leq f(E, c) \leq c$.

Definition 6.4.1 *The* constrained equal award rule (CEA) *is defined by*

$$CEA_i(E, c) = \min\{\alpha, c_i\}$$

for all problems $(E, c) \in BR^N$ *and all* $i \in N$*, where* α *is such that*
$\sum\limits_{i \in N} \min\{\alpha, c_i\} = E.$

The constrained equal award rule divides the estate as equally as possible among the players, given that no one can receive more than his claim.

Definition 6.4.2 *The* constrained equal loss rule (CEL) *is defined by*

$$CEL_i(E, c) = \max\{0, c_i - \beta\}$$

for all problems $(E, c) \in BR^N$ *and all* $i \in N$*, where* β *is such that*
$\sum\limits_{i \in N} \max\{0, c_i - \beta\} = E.$

The constrained equal loss rule divides the loss, which is the claim minus the amount received, as equally as possible among the players, given that no one can receive a negative amount. The constrained equal award rule and constrained equal loss rule are closely related, which is shown in the following well-known proposition that is readily derived.

Proposition 6.4.3 *The constrained equal award rule is the dual of the constrained equal loss rule and vice versa, i.e.,*

$$CEA(E, c) = c - CEL(\sum_{i \in N} c_i - E, c)$$

for all problems $(E, c) \in BR^N$*.*

A rule that combines the constrained equal award and constrained equal loss rule is the Aumann-Maschler rule.

Definition 6.4.4 *(cf. Aumann and Maschler (1985))*
The Aumann-Maschler rule (AM) *is defined by*

$$AM(E,c) = \begin{cases} CEA(E, \frac{1}{2}c) & \text{if } \sum_{i\in N} \frac{1}{2}c_i \geq E, \\ \frac{1}{2}c + CEL(E - \sum_{i\in N} \frac{1}{2}c_i, \frac{1}{2}c) & \text{if } \sum_{i\in N} \frac{1}{2}c_i < E, \end{cases}$$

for all problems $(E,c) \in BR^N$.

We refer to Aumann and Maschler (1985) for a motivation based on the concede and divide principle and consistency.

O'Neill (1982) associates with every problem $(E,c) \in BR^N$ a corresponding bankruptcy game $v_{E,c} \in TU^N$. In each bankruptcy game, the worth of coalition $S \in 2^N$ is the part of the estate that is left after all the players outside the coalition, *i.e.*, the players in $N \backslash S$, receive their claim. However, the worth of coalition S equals 0 when the estate is not enough to cover the amount claimed by $N \backslash S$. Formally,

$$v_{E,c}(S) = \max\{0, E - \sum_{i\in N\backslash S} c_i\} \text{ for all } S \in 2^N.$$

It is well known that every bankruptcy game has a non empty core, so one can use the propositions in Section 6.3 to characterize the (per capita) nucleolus. The nucleolus for bankruptcy games corresponds to the Aumann-Maschler rule.

Proposition 6.4.5 *(cf. Aumann and Maschler (1985))*
Let $(E,c) \in BR^N$ *and let* $v_{E,c}$ *be the corresponding bankruptcy game. Then,*

$$AM(E,c) = n(v_{E,c}).$$

Example 6.4.1 Consider the bankruptcy problem with player set $N = \{1,2,3\}$, $c = (100, 200, 300)$ and $E = 400$. Since $\sum_{i \in N} \frac{1}{2} c_i < E$, we have
$AM(E,c) = (50, 100, 150) + CEL(100, (50, 100, 150)) = (50, 125, 225)$. The corresponding bankruptcy game $v_{E,c}$ and the excesses of AM are provided below.

S	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	$\{1,2,3\}$
$v_{E,c}(S)$	0	0	100	100	200	300	400
$exc(v_{E,c}, S, AM(E,c))$	-50	-125	-125	-75	-75	-50	0

Since $\tau = 2$, $\mathcal{D}_1 = \{\{1\}, \{2,3\}\}$ and $\mathcal{D}_2 = \{\{1,2\}, \{1,3\}\}$ satisfy the conditions of Proposition 6.3.3 we have that $n(v_{E,c}) = AM(E,c) = (50, 125, 225)$. $\triangleleft$

Chapter 7 will discuss the per capita nucleolus for bankruptcy games.

Chapter 7

The per capita nucleolus and the clights rule

7.1 Introduction

This chapter, which is based on Huijink, Borm, Kleppe, and Reijnierse (2015), introduces the clights bankruptcy rule and a family of claim-and-right rules for bankruptcy problems that contains the clights rule as a specific member. The essential feature of the clights rule is that, for each bankruptcy problem, it takes into account a vector of clights, which only depends on the claim vector and not on the estate. The new term clight, a blend of claim and right, is explained below. The clights rule allocates to each claimant at most his clight when the estate is less than the sum of the clights. In this case the clights can be viewed as a modified claim vector. However, each claimant will receive at least his clight when the estate exceeds the sum of the clights. Hence, in the latter case the clights can be viewed as rights of the claimants. When the clights represent modified claims, the clights rule divides the estate over the claimants using the constrained equal award rule with the clights as new claims. Whenever the clights represent rights, the clights rule first assigns to every claimant its right. Then, the remaining estate is divided using the constrained equal loss rule with the original claims minus the clights as the new claim vector. In Section 7.2, it is shown that the clights rule coincides with the per capita nucleolus of the corresponding bankruptcy game. Furthermore, several properties of the clights rule are presented. The unified class of claim-and-right bankruptcy rules is introduced in Section 7.3. Finally, it is shown that the claim-and-right family coincides with the increasing-constant-increasing family of Thomson (2008).

7.2 Bankruptcy and the per capita nucleolus

This section introduces a new (bankruptcy) rule σ that is based on so called clights. These clights can be interpreted as either the claims of the players when the estate is relatively small or as the rights of the players when the estate is relatively large. Moreover, it is proven that this new rule coincides with the per capita nucleolus of the corresponding bankruptcy game. Throughout the remainder of this chapter we assume, for notational ease and without loss of generality, that claim vectors are weakly increasing. In other words, we assume that $c_1 \leq c_2 \leq ... \leq c_n$ for a (bankruptcy) problem (N, E, c) with $N = \{1, ..., n\}$.

Definition 7.2.1 *The* clights rule σ *is defined by*

$$\sigma(E,c) := \begin{cases} CEA(E, \delta(c)) & \text{if } \sum_{i \in N} \delta_i(c) \geq E, \\ \delta(c) + CEL(E - \sum_{i \in N} \delta_i(c), c - \delta(c)) & \text{if } \sum_{i \in N} \delta_i(c) < E, \end{cases} \tag{7.1}$$

for all problems $(E, c) \in BR^N$, *where the* clight vector $\delta(c) \in \mathbb{R}^N$ *is recursively defined for all* $i \in N$ *by*

$$\delta_i(c) := \max_{j \in \{1,...,i\}} \Big\{ \frac{1}{n+j-1} (jc_i - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \Big\} \tag{7.2}$$

Example 7.2.1 Consider a problem with player set $N = \{1, 2, 3, 4\}$ and vector of claims $c = (4, 9, 10, 19)$. Then

$$\begin{aligned} \delta_1(c) &= \frac{1}{n+1-1} \Big(1c_1 - (n-1)0 \Big) = \frac{1}{4} 4 = 1 \\ \delta_2(c) &= \max_{j \in \{1,2\}} \{ \frac{1}{n+j-1} \Big(jc_2 - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c) \Big) \} \\ &= \max\{ \frac{1}{4} \cdot 9 - \frac{3}{4} \cdot 0,\ \frac{2}{5} \cdot 9 - \frac{3}{5} \cdot 1 \} \\ &= \max\{ 2\frac{1}{4},\ 3 \} \\ &= 3, \\ \delta_3(c) &= \max\{ 2\frac{1}{2},\ 3\frac{2}{5},\ 3 \} = 3\frac{2}{5}, \end{aligned}$$

$$\delta_4(c) = \max\{4\frac{3}{4},\ 7, 7\frac{1}{2}, 7\frac{24}{35}\} = 7\frac{24}{35}.$$

Case 1: small estate ($\sum_{i\in N} \delta_i(c) \geq E$):
Consider $N = \{1,2,3,4\}$ and $c = (4,9,10,19)$ as above and take $E = 10.5$. Consequently, $\delta(c) = (1,3,3\frac{2}{5},7\frac{24}{35})$ and $\sum_{i\in N} \delta_i(c) = 15\frac{3}{35} > 10.5 = E$. In this case, the clight vector $\delta(c)$ is interpreted as the appropriate vector of claims and $\sigma(E,c) = CEA(E,\delta(c)) = (1,3,3\frac{1}{4},3\frac{1}{4})$.

Case 2: large estate ($\sum_{i\in N} \delta_i(c) < E$):
Consider $N = \{1,2,3,4\}$ and $c = (4,9,10,19)$ as above but now take $E = 20.5$. Consequently, $\delta(c) = (1,3,3\frac{2}{5},7\frac{24}{35})$. Now $\sum_{i\in N} \delta_i(c) = 15\frac{3}{35} < 20.5 = E$. In this case, the clight vector $\delta(c)$ is interpreted as the vector of rights and
$\sigma(E,c) = \delta(c) + CEL(E - \sum_{i\in N} \delta_i(c), c - \delta(c)) = (1,3,3\frac{3}{4},12\frac{3}{4})$.

A hydraulic interpretation: Suppose that the estate symbolizes an amount of water and that the claims symbolize the amount of water claimed. Then, each claim can be represented by a bucket which has the volume of that claim. In the clights rule σ, each bucket is split into two smaller buckets, namely the clights buckets of volume $\delta(c)$ and the remainder bucket of volume $c - \delta(c)$. This is visualized in Figure 7.1, where the water will be poured into the buckets at the arrow and any overspill will flood from the buckets of volume $\delta(c)$ into the buckets of volume $c - \delta(c)$.

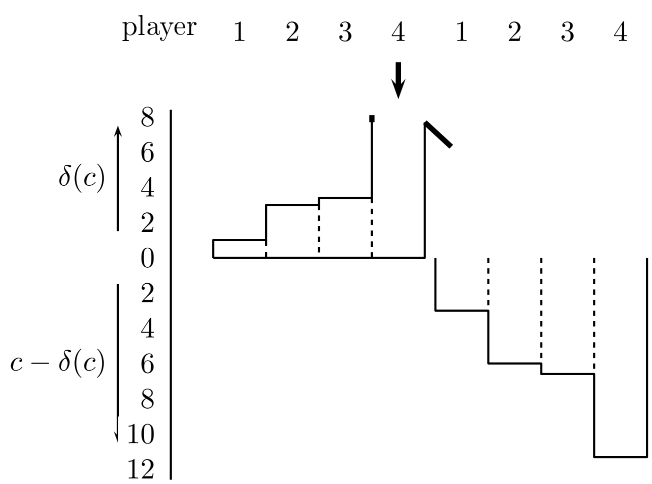

Figure 7.1: The buckets of the σ rule.

In Figure 7.2, the small estate case with $E = 10.5$ is visualized. The water is poured into the buckets at the arrow and the result is visualized by the dashed area.

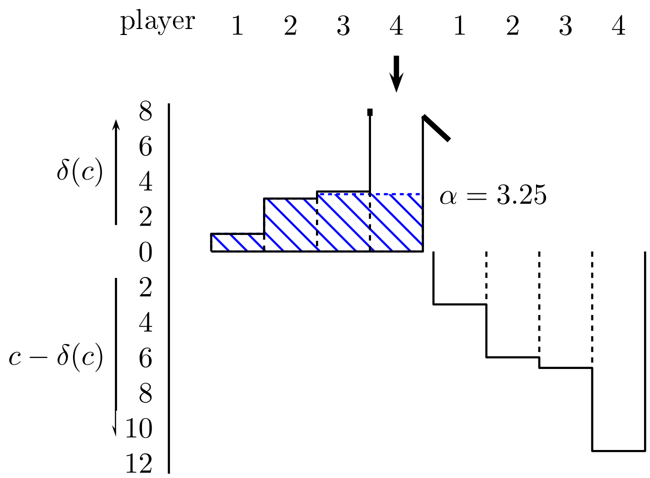

Figure 7.2: $\sigma(N, 10.5, c)$ visualized.

In Figure 7.3, the case with the large estate $E = 20.5$ is visualized. Again the water is poured into the buckets at the arrow, but now there is overflow of size $E - \sum_{i \in N} \delta(c) = 20.5 - 15\frac{3}{35} = 5\frac{29}{70}$. Again, the result is visualized by the dashed area. $\triangleleft$

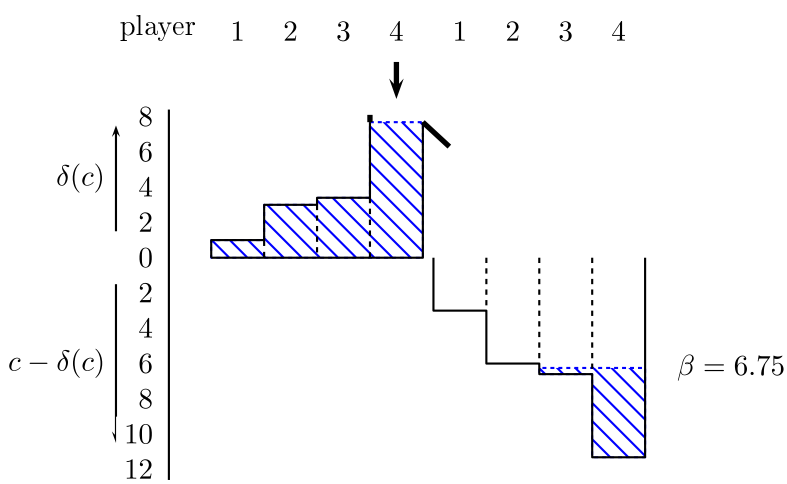

Figure 7.3: $\sigma(N, 20.5, c)$ visualized.

The following lemma implies that each clight is non-negative and is less than the claim of the corresponding player.

Lemma 7.2.2 *Let $c \in \mathbb{R}^N_+$ be a vector of claims. Then, for all $i \in N$,*

$$\frac{1}{n} c_i \leq \delta_i(c) \leq \frac{i}{n+i-1} c_i.$$

Proof: Let $i \in N$. Then

$$\delta_i(c) = \max_{j \in \{1,...,i\}} \left\{ \frac{1}{n+j-1} (jc_i - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} \geq \frac{1}{n} c_i,$$

since the right hand side corresponds to the case $j = 1$. Consequently, $\delta_j(c) \geq 0$ for all $j \in N$ and

$$\begin{aligned} \delta_i(c) &= \max_{j \in \{1,...,i\}} \left\{ \frac{1}{n+j-1} (jc_i - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} \\ &\leq \max_{j \in \{1,...,i\}} \left\{ \frac{1}{n+j-1} jc_i \right\} \\ &= \frac{i}{n+i-1} c_i, \end{aligned}$$

where the final equation is due to the fact that $\frac{i}{n+i-1} < \frac{i+1}{n+(i+1)-1}$ for all $i \in \{1, ..., n-1\}$. □

That the clights rule indeed is a bankruptcy rule, $0 \leq \sigma(E, c) \leq c$ for all bankruptcy problems (E, c), follows from Lemma 7.2.2 together with the fact that both the CEA and CEL rules satisfy the bankruptcy rule conditions. Lemma 7.2.3 shows that the clights form a non-decreasing sequence, in other words, the clights are monotonic.

Lemma 7.2.3 *Let $c \in \mathbb{R}_+^N$ be a vector of claims. Then, for all $i \in \{2, ..., n\}$,*

$$\delta_i(c) \geq \delta_{i-1}(c).$$

Proof: Let $i \in \{2, ..., n\}$. Then

$$\begin{aligned} \delta_i(c) &= \max_{j \in \{1,...,i\}} \left\{ \frac{1}{n+j-1} (jc_i - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} \\ &\geq \max_{j \in \{1,...,i-1\}} \left\{ \frac{1}{n+j-1} (jc_i - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} \\ &\geq \max_{j \in \{1,...,i-1\}} \left\{ \frac{1}{n+j-1} (jc_{i-1} - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} \\ &= \delta_{i-1}(c). \end{aligned}$$

□

The player $j \in \{1, ..., i\}$ with highest index for whom the maximum in (7.2) is attained for $\delta_i(c)$ is of importance later on. This player is called the clight-argument of player i and is formally defined below.

Definition 7.2.4 *Let $c \in \mathbb{R}^N_+$ be a vector of claims. Then, the* clight-argument *$a_i(c) \in N$ of player $i \in N$ is defined by*

$$a_i(c) := \max\Big\{j \in \{1, ..., i\} \mid \delta_i(c) = \frac{1}{n+j-1}(jc_i - (n-1)\sum_{\ell=1}^{j-1}\delta_\ell(c))\Big\} \quad (7.3)$$

Similarly as for the clights, we have monotonicity for the clight-arguments.

Lemma 7.2.5 *Let $c \in \mathbb{R}^N_+$ be a vector of claims. Then, for all $i \in \{2, ..., n\}$,*

$$a_i(c) \geq a_{i-1}(c).$$

Proof: Let $i \in \{2, ..., n\}$ and let $k \in \{1, ..., a_{i-1}(c)\}$. We show that $a_i(c) \geq k$, which completes the proof. By the definition of $a_{i-1}(c)$ we have that

$$\frac{1}{n+a_{i-1}(c)-1}(a_{i-1}(c)c_{i-1}-(n-1)\sum_{\ell=1}^{a_{i-1}(c)-1}\delta_\ell(c)) \geq \frac{1}{n+k-1}(kc_{i-1}-(n-1)\sum_{\ell=1}^{k-1}\delta_\ell(c))$$

which rewrites to

$$\Big(\frac{a_{i-1}(c)}{n+a_{i-1}(c)-1}-\frac{k}{n+k-1}\Big)c_{i-1} \geq \frac{n-1}{n+a_{i-1}(c)-1}\sum_{\ell=1}^{a_{i-1}(c)-1}\delta_\ell(c)-\frac{n-1}{n+k-1}\sum_{\ell=1}^{k-1}\delta_\ell(c).$$

Since $c_i \geq c_{i-1}$ and $\frac{a_{i-1}(c)}{n+a_{i-1}(c)-1} \geq \frac{k}{n+k-1}$ we have

$$\Big(\frac{a_{i-1}(c)}{n+a_{i-1}(c)-1}-\frac{k}{n+k-1}\Big)c_i \geq \Big(\frac{a_{i-1}(c)}{n+a_{i-1}(c)-1}-\frac{k}{n+k-1}\Big)c_{i-1}.$$

Hence,

$$\Big(\frac{a_{i-1}(c)}{n+a_{i-1}(c)-1}-\frac{k}{n+k-1}\Big)c_i \geq \frac{n-1}{n+a_{i-1}(c)-1}\sum_{\ell=1}^{a_{i-1}(c)-1}\delta_\ell(c)-\frac{n-1}{n+k-1}\sum_{\ell=1}^{k-1}\delta_\ell(c).$$

which can be written as

$$\frac{1}{n+a_{i-1}(c)-1}(a_{i-1}(c)c_i-(n-1)\sum_{\ell=1}^{a_{i-1}(c)-1}\delta_\ell(c))\geq\frac{1}{n+k-1}(kc_i-(n-1)\sum_{\ell=1}^{k-1}\delta_\ell(c)).$$

Hence, $a_i(c) \geq a_{i-1}(c)$. □

In the following proposition, it is shown that the clight-arguments of all players except for player 1 cannot be player 1 and it provides an explicit expression for the clight vector of two- and three-player problems.

Proposition 7.2.6 *Let $c \in \mathbb{R}^N_+$ be a vector of claims. Then, for all $i \in \{2, ..., n\}$,*

$$a_i(c) \geq 2.$$

Furthermore,

$$\delta(c) = \begin{cases} (\frac{1}{2}c_1, \frac{2}{3}c_2 - \frac{1}{3}c_1) & \text{if } N = \{1,2\}, \\ (\frac{1}{3}c_1, \frac{1}{2}c_2 - \frac{1}{6}c_1, \max\{\frac{1}{2}c_3 - \frac{1}{6}c_1, \frac{3}{5}c_3 - \frac{1}{5}c_2 - \frac{1}{15}c_1\}) & \text{if } N = \{1,2,3\}. \end{cases}$$

Proof: Since $a_i(c) \geq a_{i-1}(c)$ (Lemma 7.2.5), we only have to prove that $a_2(c) \geq 2$. By definition, we have that $\delta_1(c) = \frac{1}{n}c_1$. Note that

$$\delta_2(c) = \max\{\frac{1}{n}c_2, \frac{1}{n+2-1}(2c_2 - (n-1)\frac{1}{n}c_1)\}.$$

Since

$$\frac{1}{n+2-1}(2c_2 - (n-1)\frac{1}{n}c_1) \geq \frac{1}{n+1}(2c_2 - \frac{n-1}{n}c_2) = \frac{1}{n}c_2,$$

we obtain $a_2(c) \geq 2$. □

The clights do not only satisfy monotonicity, but also monotonicity of losses. In other words, the claims minus the clights form a non-decreasing sequence.

Lemma 7.2.7 *Let $c \in \mathbb{R}^N_+$ be a vector of claims. Then, for all $i \in \{2, ..., n\}$,*

$$\delta_i(c) - \delta_{i-1}(c) \leq \frac{i}{n+i-1}(c_i - c_{i-1}).$$

Hence, $c_{i-1} - \delta_{i-1}(c) \leq c_i - \delta_i(c)$ for all $i \in \{2, ..., n\}$.

Proof: Let $i \in \{2, ..., n\}$. The proof is split into two cases, depending on the clight-argument.

Case 1: Assume $a_i(c) \leq i-1$. Then

$$\begin{aligned}
\delta_i(c) - \delta_{i-1}(c) &= \frac{1}{n + a_i(c) - 1}(a_i(c)c_i - (n-1)\sum_{\ell=1}^{a_i(c)-1} \delta_\ell(c)) - \delta_{i-1}(c) \\
&\leq \frac{1}{n + a_i(c) - 1}(a_i(c)c_i - (n-1)\sum_{\ell=1}^{a_i(c)-1} \delta_\ell(c)) \\
&\qquad - \frac{1}{n + a_i(c) - 1}(a_i(c)c_{i-1} - (n-1)\sum_{\ell=1}^{a_i(c)-1} \delta_\ell(c)) \\
&= \frac{a_i(c)}{n + a_i(c) - 1}(c_i - c_{i-1}) \\
&\leq \frac{i}{n+i-1}(c_i - c_{i-1}).
\end{aligned}$$

Case 2: Assume that $a_i(c) = i$. Then,

$$\begin{aligned}
\delta_i(c) - \delta_{i-1}(c) &= \frac{1}{n+i-1}(ic_i - (n-1)\sum_{\ell=1}^{i-1} \delta_\ell(c)) - \delta_{i-1}(c) \\
&= \frac{1}{n+i-1}(ic_i - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c)) - \frac{n-1}{n+i-1}\delta_{i-1}(c) - \delta_{i-1}(c) \\
&= \frac{1}{n+i-1}(ic_i - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c)) - \frac{n}{n+i-1}\delta_{i-1}(c) - \frac{n+i-2}{n+i-1}\delta_{i-1}(c) \\
&\leq \frac{1}{n+i-1}(ic_i - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c)) - \frac{1}{n+i-1}c_{i-1} - \frac{n+i-2}{n+i-1}\delta_{i-1}(c) \\
&\leq \frac{1}{n+i-1}(ic_i - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c)) - \frac{1}{n+i-1}c_{i-1} \\
&\qquad - \frac{n+i-2}{n+i-1}\Big(\frac{1}{n+(i-1)-1}((i-1)c_{i-1} - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c))\Big) \\
&= \frac{1}{n+i-1}(ic_i - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c)) - \frac{1}{n+i-1}c_{i-1} \\
&\qquad - \Big(\frac{1}{n+i-1}((i-1)c_{i-1} - (n-1)\sum_{\ell=1}^{i-2} \delta_\ell(c))\Big) \\
&= \frac{i}{n+i-1}(c_i - c_{i-1})
\end{aligned}$$

where the first inequality follows from the fact that $\delta_{i-1}(c) \geq \frac{1}{n}c_{i-1}$ (Lemma 7.2.2). □

The next example acts as a stepping stone for the proof that the clights rule coincides with the per capita nucleolus of a bankruptcy game.

Example 7.2.2 (Example 7.2.1 continued.)
Case 1: small estate, i.e., $\sum_{i \in N} \delta_i(c) \geq E$:
Consider the problem (E, c) with $N = \{1,2,3,4\}$, $E = 10.5$ and $c = (4, 9, 10, 19)$. As we have seen, $\delta(c) = (1, 3, 3\frac{2}{5}, 7\frac{24}{35})$, $\sigma(E, c) = (1, 3, 3\frac{1}{4}, 3\frac{1}{4})$, $a_1(c) = 1$, $a_2(c) = a_3(c) = 2$ and $a_4(c) = 4$. The corresponding bankruptcy game and the per capita excesses of $\sigma(E, c)$ are as follows:

S	$\{1\}$	$\{2\}$	$\{3\}$	$\{4\}$	$\{1,2\}$	$\{1,3\}$	$\{1,4\}$	$\{2,3\}$
$v_{E,c}(S)$	0	0	0	0	0	0	0	0
$exc^P(v_{E,c}, S, \sigma(E,c))$	-1	-3	$-3\frac{1}{4}$	$-3\frac{1}{4}$	-2	$-2\frac{1}{8}$	$-2\frac{1}{8}$	$-3\frac{1}{8}$

S		$\{2,4\}$	$\{3,4\}$	$\{1,2,3\}$	$\{1,2,4\}$	$\{1,3,4\}$	$\{2,3,4\}$	$\{1,2,3,4\}$
$v_{E,c}(S)$		0	0	0	0.5	1.5	6.5	10.5
$exc^P(vE, c, S, \sigma(E,c))$		$-3\frac{1}{8}$	$-3\frac{1}{4}$	$-2\frac{5}{12}$	$-2\frac{1}{4}$	-2	-1	0

Set $\mathcal{D}_1 = \{\{1\},\{2,3,4\}\}$, $\mathcal{D}_2 = \{\{1,2\},\{1,3,4\}\}$ and $\mathcal{D}_3 = \{\{1,3\},\{1,4\}\}$. Then, $\mathcal{D}_1 \subseteq 2^N \backslash \{\varnothing, N\} = \mathcal{F}(\{\varnothing\})$ and clearly $\bar{\mathcal{D}}_1 = \{\{1\},\{2,3,4\}\}$ is balanced. Furthermore, the coalitions in $\mathcal{D}_1$ have the highest per capita excess of all (free) coalitions. Similarly, $\mathcal{D}_2 \subseteq 2^N \backslash \{\varnothing, \{1\}, \{2,3,4\}, N\} = \mathcal{F}(\bar{\mathcal{D}}_1)$ and $\bar{\mathcal{D}}_2 = \{\{1\},\{2,3,4\},\{1,2\},\{1,3,4\}\}$ is balanced since $\rho = (\frac{1}{3}, \frac{2}{3}, \frac{1}{3}, \frac{1}{3})$, which assigns to the corresponding coalitions in $\bar{\mathcal{D}}_2$ a weight, is a balanced map for $\bar{\mathcal{D}}_2$. The coalitions in $\mathcal{D}_2$ have the highest per capita excess of all currently free coalitions, where the currently free coalitions are $\mathcal{F}(\bar{\mathcal{D}}_1)$. Furthermore,
$\mathcal{D}_3 \subseteq \{\{3\},\{4\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{1,2,3\},\{1,2,4\}\} = \mathcal{F}(\bar{\mathcal{D}}_2)$ and
$\bar{\mathcal{D}}_3 = \{\{1\},\{2,3,4\},\{1,2\},\{1,3,4\},\{1,3\},\{1,4\}\}$ is balanced since
$\rho = (\frac{1}{6}, \frac{4}{6}, \frac{2}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$ is a corresponding balanced map. The coalitions in $\mathcal{D}_3$ have the highest per capita excess of all currently free coalitions ($\mathcal{F}(\bar{\mathcal{D}}_2)$). Finally, note that $\mathcal{F}(\bar{\mathcal{D}}_3) = \varnothing$. Using Proposition 6.3.4, we conclude that $\sigma(E, c) = pcn(v_{E,c})$.

Case 2: large estate, i.e., $\sum_{i \in N} \delta_i(c) < E$:
Consider the problem (E, c) with $N = \{1,2,3,4\}$, $E = 20.5$ and $c = (4, 9, 10, 19)$.

As we have seen, $\delta(c) = (1, 3, 3\frac{2}{5}, 7\frac{24}{35})$, $\sigma(E, c) = (1, 3, 3\frac{3}{4}, 12\frac{3}{4})$, $a_1(c) = 1$, $a_2(c) = a_3(c) = 2$ and $a_4(c) = 4$. The corresponding bankruptcy game and the per capita excesses of $\sigma(E, c)$ are as follows:

S	$\{1\}$	$\{2\}$	$\{3\}$	$\{4\}$	$\{1,2\}$	$\{1,3\}$	$\{1,4\}$	$\{2,3\}$
$v_{E,c}$	0	0	0	0	0	0	1.5	0
$exc^P(v_{E,c}, S, \sigma(E,c))$	-1	-3	$-3\frac{3}{4}$	$-12\frac{3}{4}$	-2	$-2\frac{3}{8}$	$-6\frac{1}{8}$	$-3\frac{3}{8}$

S		$\{2,4\}$	$\{3,4\}$	$\{1,2,3\}$	$\{1,2,4\}$	$\{1,3,4\}$	$\{2,3,4\}$	$\{1,2,3,4\}$
$v_{E,c}$		6.5	7.5	1.5	10.5	11.5	16.5	20.5
$exc^P(v_{E,c}, S, \sigma(E,c))$		$-4\frac{5}{8}$	$-4\frac{1}{2}$	$-2\frac{1}{12}$	$-2\frac{1}{12}$	-2	-1	0

Now, take $\mathcal{D}_1 = \{\{1\}, \{2,3,4\}\}$, $\mathcal{D}_2 = \{\{1,2\}, \{1,3,4\}\}$ and $\mathcal{D}_3 = \{\{1,2,4\}, \{1,2,3\}\}$. Regarding $\mathcal{D}_1$ and $\mathcal{D}_2$ we refer to case 1. Moreover, $\mathcal{D}_3 \subseteq \{\{3\}, \{4\}, \{1,3\}, \{1,4\}, \{2,3\}, \{2,4\}, \{1,2,3\}, \{1,2,4\}\} = \mathcal{F}(\bar{\mathcal{D}}_2)$ and $\bar{\mathcal{D}}_3 = \{\{1\}, \{2,3,4\}, \{1,2\}, \{1,3,4\}, \{1,2,4\}, \{1,2,3\}\}$ is balanced since $\rho = (\frac{1}{6}, \frac{3}{6}, \frac{1}{6}, \frac{2}{6}, \frac{1}{6}, \frac{1}{6})$ is a corresponding balanced map. The coalitions in $\mathcal{D}_3$ have the highest per capita excess of all the currently free coalitions. Finally, note that $\mathcal{F}(\bar{\mathcal{D}}_3) = \varnothing$. Again, using Proposition 6.3.4 we can conclude that $\sigma(E, c) = pcn(v_{E,c})$.

General remarks on the cases:
The relevant collections of these examples can be expressed by the clight-argument vector. For both cases ($E = 10.5$ and $E = 20.5$) we have that

$$\mathcal{D}_1 = \{\{1\}, \{2,3,4\}\} = \{\{1, ..., a_1(c) - 1\} \cup \{1\}, N\backslash\{1\}\},$$
$$\mathcal{D}_2 = \{\{1,2\}, \{1,3,4\}\} = \{\{1, ..., a_2(c) - 1\} \cup \{2\}, N\backslash\{2\}\}.$$

Furthermore, when $E = 10.5$, we have that

$$\mathcal{D}_3 = \{\{1,3\}, \{1,4\}\} = \{\{1, ..., a_3(c) - 1\} \cup \{3\}, \{1, ..., a_4(c) - 1\} \cup \{4\}\},$$

and when $E = 20.5$, we have that

$$\mathcal{D}_3 = \{\{1,3,4\}, \{1,2,3\}\} = \{N\backslash\{3\}, N\backslash\{4\}\}.$$

This structure of the relevant collections will form the basis of the proof of our main result. ◁

Theorem 7.2.8 *Let $(E, c) \in BR^N$ and let $v_{E,c}$ be the corresponding bankruptcy game. Then,*

$$\sigma(E, c) = pcn(v_{E,c}).$$

Proof: In this proof, $\sigma(E, c)$ is abbreviated to σ and $v_{E,c}$ is abbreviated to v. Furthermore, since σ and the per capita nucleolus both depend continuously on the estate E, we assume that $\sum_{\ell \in N} \delta_\ell(c) \neq E$.

In order to apply Proposition 6.3.4, we will do the following:
Part I: Define τ, and for all $r \in \{1, ..., \tau\}$, define appropriate relevant collections $\mathcal{D}_r$ and show that $\mathcal{D}_r \subset \mathcal{F}(\bar{\mathcal{D}}_{r-1})$.
Part II: Show that the sequence $\mathcal{D}_1, ..., \mathcal{D}_\tau$ satisfies condition (A) of Proposition 6.3.4, *i.e.*, $\bar{\mathcal{D}}_r = \bigcup_{\ell=1}^{r} \mathcal{D}_\ell$ is balanced for all $r \in \{1, ..., \tau\}$ and $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$.
Part III: Show that the sequence $\mathcal{D}_1, ..., \mathcal{D}_\tau$ satisfies condition (B) of Proposition 6.3.4, *i.e.*, for all $r \in \{1, ..., \tau\}$ and all $S \in \mathcal{D}_r$, it holds that
$exc^P(v, S, \sigma) = \max\limits_{T \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})} exc^P(v, T, \sigma)$.

The remainder of the proof can be found in Appendix E $\square$

In the proof of Theorem 7.2.8, it is seen that the relevant coalitions of the per capita nucleolus have a special structure for bankruptcy games. This is formalized in the following corollary.

Corollary 7.2.9 *Let $(E, c) \in BR^N$ be a problem and let $x = pcn(v_{E,c})$. Then, there exists a player $t \in \{1, ..., n\}$ and a sequence $\mathcal{D}_1, \ldots, \mathcal{D}_\tau \subset 2^N \backslash \{\varnothing, N\}$, where $\tau = \min\{t, n-1\}$, that satisfy (A) and (B) of Proposition 6.3.4, where*

$$\mathcal{D}_r = \{\{1, ..., a_r(c) - 1\} \cup \{r\}, N \backslash \{r\}\},$$

if $r < \tau$. Furthermore,

$$\mathcal{D}_\tau = \begin{cases} \{\{1, ..., a(\tau) - 1\} \cup \{\tau\}, N \backslash \{\tau\}\} & \text{if } \tau < t, \\ \{\{1, ..., m-1\} \cup \{\tau\}, ..., \{1, ..., m-1\} \cup \{n\}\} & \text{if } \tau = t \text{ and } \sum\limits_{\ell \in N} \delta_\ell(c) > E, \\ \{N \backslash \{\tau\}, ..., N \backslash \{n\}\} & \text{if } \tau = t \text{ and } \sum\limits_{\ell \in N} \delta_\ell(c) < E, \end{cases}$$

for some $m \in \{1, ..., \tau\}$.

Several properties that the clights rule does or does not satisfy are provided below. Proposition 7.2.19 provides a compact overview of these results.

Definition 7.2.10 *A rule satisfies* order preservation *if for all* $(E,c) \in BR^N$ *and all* $i \in \{1,...,n-1\}$, *it holds that* $f_i(E,c) \leq f_{i+1}(E,c)$ *and* $c_i - f_i(E,c) \leq c_{i+1} - f_{i+1}(E,c)$.

A rule satisfies anonymity *if for all* $(E,c) \in BR^N$, *any permutation* π, *and all* $i \in N$, *it holds that* $f_i(E,c) = f_{\pi(i)}(E,(c_{\pi(j)})_{j\in N})$.

A rule satisfies equal treatment of equals *if for all* $(E,c) \in BR^N$ *and all* $i,j \in N$ *with* $c_i = c_j$, *it holds that* $f_i(E,c) = f_j(E,c)$.

Definition 7.2.11 *A rule satisfies* claims truncation invariance *if for all* $(E,c) \in BR^N$, *it holds that* $f(E,c) = f(E,(\min\{c_j,E\})_{j\in N})$.

A rule satisfies resource monotonicity *if for all* $(E,c) \in BR^N$ *and all* $x \in (0, \sum_{i\in N} c_i - E]$, *it holds that* $f(E,c) \leq f(E+x,c)$.

A rule satisfies minimal rights first *if for all* $(E,c) \in BR^N$, *it holds that* $f(E,c) = m + f(E - \sum_{i\in N} m_i, c-m)$ *where* $m = (\max\{E - \sum_{j\in N\backslash\{i\}} c_j, 0\})_{i\in N}$.

A rule satisfies respect of minimal rights *if for all* $(E,c) \in BR^N$ *and all* $i \in N$, *it holds that* $f_i(E,c) \geq \max\{E - \sum_{j\in N\backslash\{i\}} c_j, 0\}$.

Definition 7.2.12 *A rule satisfies the* concede-and-divide principle *if for all* $(E,c) \in BR^N$ *with* $|N| = 2$ *and all* $i \in N$, *it holds that* $f_i(E,c) = \max\{E - c_{N\backslash\{i\}}, 0\} + \frac{1}{2}(E - \sum_{\ell\in N} \max\{E - c_{N\backslash\{\ell\}}, 0\})$.

A rule satisfies claims monotonicity *if for all* $(E,c) \in BR^N$ *and all* $i \in N$, *it holds that* $f_i(E,\bar{c}) \geq f_i(E,c)$, *whenever* $\bar{c}_i > c_i$ *and* $\bar{c}_j = c_j$ *for all* $j \in N\backslash\{i\}$.

A rule satisfies linked resource-claims monotonicity *if for all* $(E,c) \in BR^N$, *all* $i \in N$, *and all* $x \in \mathbb{R}_+$, *it holds that* $f_i(E+x,\bar{c}) \leq f_i(E,c) + x$, *whenever* $\bar{c}_i = c_i + x$ *and* $\bar{c}_j = c_j$ *for all* $j \in N\backslash\{i\}$.

Definition 7.2.13 *A rule satisfies* regressivity *if for all* $(E,c) \in BR^N$ *and all* $i \in \{1,...,n-1\}$ *with* $c_i > 0$, *it holds that* $\frac{f_i(E,c)}{c_i} \leq \frac{f_{i+1}(E,c)}{c_{i+1}}$.

A rule satisfies progressivity *if for all* $(E,c) \in BR^N$ *and for all* $i \in \{1,...,n-1\}$ *with* $c_i > 0$*, it holds that* $\frac{f_i(E,c)}{c_i} \geq \frac{f_{i+1}(E,c)}{c_{i+1}}$.

Definition 7.2.14 *A rule satisfies* composition down *if for all* $(E,c) \in BR^N$ *and all* $x \in (0,E]$*, it holds that* $f(E-x,c) = f(E-x, f(c,E))$.

A rule satisfies composition up *if for all* $(E,c) \in BR^N$ *and all* $x \in (0,E]$*, it holds that* $f(E,c) = f(E-x,c) + f(x, c - f(c, E-x))$.

Definition 7.2.15 *A rule satisfies* no advantageous transfer *if for all* $(E,c) \in BR^N$*, all* $\bar{N} \subset N$*, and each* $\bar{c} \in \mathbb{R}^N_+$ *such that* $\sum_{i \in \bar{N}} \bar{c}_i = \sum_{i \in \bar{N}} c_i$ *and* $\bar{c}_i = c_i$ *for all* $i \in N \backslash \bar{N}$*, it holds that* $\sum_{i \in \bar{N}} f_i(E,\bar{c}) = \sum_{i \in \bar{N}} f_i(E,c)$.

Definition 7.2.16 *A rule satisfies* self-duality *if for all* $(E,c) \in BR^N$*, it holds that* $f(E,c) = c - f(\sum_{i \in N} c_i - E, c)$.

For the definitions of consistency, null-claims consistency and converse consistency, we refer to Thomson (2003).

Definition 7.2.17 *A rule satisfies* others oriented claims monotonicity *if for all* $(E,c) \in BR^N$*, all* $i \in N$*, and all* $j \in N \backslash \{i\}$*, it holds that* $f_i(E,\bar{c}) \geq f_i(E,c)$ *and* $f_j(E,\bar{c}) \leq f_j(E,c)$*, whenever* $\bar{c}_i > c_i$ *and* $\bar{c}_j = c_j$ *for all* $j \in N \backslash \{i\}$.

Definition 7.2.18 *A rule satisfies* population monotonicity *if for all* $(E,c) \in BR^N$*, and all* $\bar{N} \subset N$ *with* $\sum_{\ell \in \bar{N}} c_\ell \geq E$ *it holds that* $f_i(N,E,c) \leq f_i(\bar{N},E,c_{\bar{N}})$ *for all* $i \in \bar{N}$*, where* $c_{\bar{N}}$ *is the vector of original claims* c_i *of the players* $i \in \bar{N}$.

A rule satisfies linked resource-population monotonicity *if for all* $(E,c) \in BR^N$ *and all* $\bar{N} \subset N$ *with* $\sum_{\ell \in \bar{N}} c_\ell \geq E$*, it holds that* $f_i(\bar{N}, E - \sum_{j \in N \backslash \bar{N}} c_j, c_{\bar{N}}) \leq f_i(N,E,c)$ *for all* $i \in \bar{N}$*, where* $c_{\bar{N}}$ *is the vector of original claims* c_i *of the players* $i \in \bar{N}$.

Proposition 7.2.19 *The clights rule satisfies: order preservation, anonymity, equal treatment of equals, claims truncation invariance, resource monotonicity, minimal rights first, respect of minimal rights, concede-and-divide principle, claims monotonicity and linked resource-claims monotonicity.*

It does not satisfy: regressivity, progressivity, composition down, composition up, no advantageous transfer, self-duality, consistency, null claims consistency, converse consistency, others oriented claims monotonicity, population monotonicity, and linked resource-population monotonicity.

Proof: Below we present the main arguments behind the proofs for the properties that the clights rule satisfies or a counterexample for the properties that it does not satisfy.

- The clights satisfy a property similar to order preservation, i.e., $\delta_{i-1}(c) \leq \delta_i(c)$ and $c_{i-1} - \delta_{i-1}(c) \leq c_i - \delta_i(c)$ for all $i \in \{2, ..., n\}$ (Lemma 7.2.3 and Lemma 7.2.7), and are bounded, in other words, $0 \leq \delta_i(c) \leq c_i$ for all $i \in N$ (Lemma 7.2.2). Since CEA and CEL both satisfy order preservation and since the clights rule is a proper combination, it satisfies order preservation.

- Anonymity and equal treatment of equals follow from the fact that the clights rule corresponds to the per capita nucleolus (Theorem 7.2.8).

- Claims truncation invariance follows from the fact that the clights rule is a game theoretic rule (as implied by Theorem 7.2.8).

- Resource monotonicity is obvious since $\delta(c)$ is independent of E and since the rule is a combination of CEA and CEL.

- Minimal rights first follows from the fact that the minimal right of player i is equal to the worth of that player, in other words, $m_i(E, c) = v_{E,c}(\{i\})$, Theorem 7.2.8 and the fact that the per capita nucleolus satisfies covariance, in other words, $pcn(v_{E,c}) = m(E, c) + pcn(v_{E,c-m})$.

- Respect of minimal rights is implied by efficiency, non-negativity, and minimal rights first.

- Concede-and-divide principle follows from Theorem 7.2.8 and the fact that $pcn(v) = n(v)$ for all TU-games (N, v) with $|N| = 2$, and hence
$AM(E, c) = \sigma(E, c)$ if $|N| = 2$.

- Claims monotonicity and its dual, linked resource-claims monotonicity, are proven to be valid simultaneously. From here, abbreviate $\sigma(E, c)$ to $\sigma(c)$. Let $(E, c) \in BR^N$, let $k \in N$ and define $\bar{c} \in \mathbb{R}^N$ by $\bar{c}_k = c_k + \varepsilon$, where $\varepsilon > 0$ and $\bar{c}_j = c_j$ for all $j \in N \backslash \{k\}$. We will prove that $\sigma_k(\bar{c}) \in [\sigma_k(c), \sigma_k(c) + \varepsilon]$.

  Due to the continuity of σ with respect to c, we can assume without loss of generality, that ε is such that $\bar{c}_1 \leq \bar{c}_2 \leq \cdots \leq \bar{c}_n$. This is true due to the fact that, in the case of equal claims, we can renumber the players such that player k becomes the last player in his "claim category". Due to continuity again, we only have to prove the assertions for a dense set of claim vectors. Therefore, we can assume that $\sum_{i \in N} \delta_i(c) \neq E$. Indeed, if $\sum_{i \in N} \delta_i(c) = E$, we can approximate the claim vector c by the sequence of claim vectors $c^1, c^2, c^3, \ldots$ with $\sum_{i \in N} \delta_i(c^\ell) > E$ for all $\ell \in \mathbb{N}$ by defining $c_n^\ell = c_n + \frac{1}{\ell}$ and $c_j^\ell = c_j$ for all $j < n$.

  Define $t(E, c) := \min\{i \in N \mid \sigma_i(c) \neq \delta_i(c)\}$ (since $\sum_{i \in N} \delta_i(c) \neq E$ such agents exist). Once more, by continuity, we can choose ε sufficiently small to establish that $t(E, c) = t(E, \bar{c})$, so we abbreviate $t(E, c)$ and $t(E, \bar{c})$ to t. We choose ε also sufficiently small to establish that $\sum_{i \in N} \delta_i(c) - E$ and $\sum_{i \in N} \delta_i(\bar{c}) - \bar{E}$ are both positive or both negative.

  We will first show that

  $$\delta_k(c) < \bar{\delta}_k(\bar{c}) < \delta_k(c) + \varepsilon. \tag{7.4}$$

  By definition, we have, for all $i \in N$,

  $\delta_i(c) = \max_{j \in \{1, \ldots, i\}} \left\{ \frac{1}{n+j-1} (jc_i - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\}$. For $i < k$, $\delta_i(c)$ does not depend on c_k, so $\delta_i(\bar{c}) = \delta_i(c)$ for all $i < k$. Therefore,

  $$\delta_k(\bar{c}) = \max_{j \in \{1, \ldots, k\}} \left\{ \frac{1}{n+j-1} (j(c_k + \varepsilon) - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} > \delta_k(c)$$

  and

  $$\begin{aligned} \delta_k(\bar{c}) &\leq \max_{j \in \{1, \ldots, k\}} \left\{ \frac{1}{n+j-1} (jc_k - (n-1) \sum_{\ell=1}^{j-1} \delta_\ell(c)) \right\} + \max_{j \in \{1, \ldots, k\}} \frac{1}{n+j-1} j\varepsilon \\ &< \delta_k(c) + \varepsilon, \end{aligned}$$

  which yields (7.4).

Recall that the clights rule allocates the estate in the following way:

$$\sigma_i(c) = \begin{cases} \delta_i(c) & \text{if } i < t, \\ \alpha(c) & \text{if } i \geq t \text{ and } \sum\limits_{\ell \in N} \delta_\ell(c) > E, \\ c_i - \beta(c) & \text{if } i \geq t \text{ and } \sum\limits_{\ell \in N} \delta_\ell(c) < E, \end{cases}$$

in which $\alpha(c)$ and $\beta(c)$ are determined by $\sum_{\ell \in N} \min\{\alpha(c), \delta_\ell(c)\} = E$ and $E - \sum_{\ell \in N} \delta_\ell(c) = \sum_{\ell \in N} \max\{0, c_\ell - \delta_\ell(c) - \beta(c)\}$, respectively.

If $k < t$, then, $\sigma_k(\bar{c}) = \delta_k(\bar{c})$ and $\sigma_k(c) = \delta_k(c)$, so $\sigma_k(\bar{c}) \in [\sigma_k(c), \sigma_k(c) + \varepsilon]$ follows from (7.4).

If $k \geq t$ and $\sum_{i \in N} \delta_i(c) > E$, then, $\delta_i(\bar{c}) = \delta_i(c)$ for all $i < t$, so

$$\sigma_k(\bar{c}) = \alpha(\bar{c}) = \alpha(c) + \frac{\varepsilon}{n - t + 1} = \sigma_k(c) + \frac{\varepsilon}{n - t + 1}.$$

Therefore, we assume from here on that $k \geq t$ and $\sum_{i \in N} \delta_i(c) < E$. We first prove that $\beta(\bar{c}) = \beta(c)$. Since

$$\begin{aligned} \sum_{i=1}^{t-1} \delta_i(c) + \sum_{i=t}^{n} (c_i - \beta(c)) = (E + \varepsilon) - \varepsilon & \\ &= \sum_{i=1}^{t-1} \delta_i(\bar{c}) + \sum_{i=t}^{n} (\bar{c}_i - \beta(\bar{c})) - \varepsilon \\ &= \sum_{i=1}^{t-1} \delta_i(c) + \sum_{i=t}^{n} (\bar{c}_i - \beta(\bar{c})) - \varepsilon, \end{aligned}$$

we have that $\sum\limits_{i=t}^{n} \beta(\bar{c}) = \sum\limits_{i=t}^{n} (\beta(c) + \bar{c}_i - c_i) - \varepsilon$ and hence $\beta(\bar{c}) = \beta(c)$. Therefore,

$$\sigma_k(\bar{c}) = \bar{c}_k - \beta(\bar{c}) = c_k + \varepsilon - \beta(c) = \sigma_k(c) + \varepsilon.$$

- In order to show that the clights rule does not satisfy regressivity and progressivity, we refer to the various problems in Example 7.2.1.
- In order to show that the clights rule does not satisfy composition down, consider the problems with $n = 2$, $c = (2, 4)$, $E = 2.5$ and $E' = 1.25$.

- In order to show that the clights rule does not satisfy composition up, consider the problems with $n = 2$, $c = (2, 4)$, $E = 1.25$ and $E' = 2.5$.

- In order to show that the clights rule does not satisfy no advantageous transfer, consider the problems with $n = 3$, $E = 300$, $c = (100, 200, 300)$ and $c' = (0, 300, 300)$.

- In order to show that the clights rule does not satisfy self-duality, consider the problem with $n = 3$, $E = 200$, and $c = (100, 200, 300)$.

- The clights rule does not satisfy consistency and converse consistency due to the fact that it is not the AM rule.

- Others oriented claims monotonicity (these problems are also used for other properties below): Let $N = \{1, 2, 3, 4\}, \bar{N} = \{2, 3, 4\}, E = 8$, $c = (1, 5, 6, 8)$ and $\bar{c} = (1\frac{1}{10}, 5, 6, 8)$. This gives $\sigma(N, E, c) = (\frac{1}{4}, 1\frac{17}{20}, 2\frac{1}{4}, 3\frac{13}{20})$ and $\sigma(N, E, \bar{c}) = (\frac{11}{40}, 1\frac{167}{200}, 2\frac{47}{200}, 3\frac{131}{200})$. So, $f_4(N, E, \bar{c}) > f_4(N, E, c)$, contradicting others oriented claims monotonicity.

- Furthermore, $\sigma(\bar{N}, E, c_{\bar{N}}) = (1\frac{2}{3}, 2\frac{1}{6}, 4\frac{1}{6})$, so $f_2(N, E, c) > f_2(\bar{N}, E, c_{\bar{N}})$, contradicting population monotonicity.

- Moreover, $\sigma(\{1, 3, 4\}, E - c_2, c_{\{1,3,4\}}) = (\frac{1}{3}, 1\frac{1}{3}, 1\frac{1}{3})$, which contradicts linked resource-monotonicity since $f_1(\{1, 3, 4\}, E - c_2, c_{\{1,3,4\}}) > f_1(N, E, c)$.

- Finally, $\sigma(N, E, (0, 5, 6, 8)) = (0, 2, 3, 3)$, which contradicts null claims consistency since $f_2(N, E, (0, 5, 6, 8)) > f_2(\bar{N}, E, c_{\bar{N}})$.

□

7.3 The claim-and-right family of bankruptcy rules

Both the Aumann-Maschler rule and the clights rule have two different regimes depending on the size of the estate. For the Aumann-Maschler rule, the estate is considered to be small if the estate is less than half of the total amount claimed and large otherwise. Hence, half of the sum of the claims can be seen as a switch-point for the Aumann-Maschler rule. Moreover, each player receives at most half of his claim in the first regime. Therefore, half of his claim can be seen as his modified claim. On the other hand, in the second regime, half of the claim is considered to be his right, since each player will receive at least half of his claim.

The clights rule has a similar setup. Namely, the estate is considered to be small if the estate is less than the total amount of the clights and the estate is large otherwise. Hence, the switch-point for the clights rule is the sum of the clights. Figure 7.4, on page 106, illustrates the clights rule. Similar to the Aumann-Maschler rule, the clights act as new claims in the first regime and rights in the second regime. Note that in both the Aumann-Maschler rule and the clights rule the constraint equal award rule is used in the first regime and the constraint equal loss rule in the second.

To show that both the Aumann-Maschler rule and the clights rule are based on the same conceptual idea, we use the concept of claim-and-right functions, which are formalized below. Let $\mathcal{C} \subseteq \mathbb{R}^N_+$ be the set of weakly increasing (claim) vectors.

Definition 7.3.1 *A function* $\lambda : \mathcal{C} \to \mathcal{C}$ *is called a* claim-and-right function *if* $c - \lambda(c) \in \mathcal{C}$ *for all* $c \in \mathcal{C}$. *The class of claim-and-right functions is denoted by* Λ.

Note that the functions $\lambda(c) = 0$, $\lambda(c) = c$, $\lambda(c) = \frac{1}{2}c$ and $\lambda(c) = \delta(c)$, for all $c \in C$, are all claim-and-right functions.

With each claim-and-right function, one can define a rule that is based on two regimes. The first regime occurs when the estate is insufficient to cover $\lambda(c)$. In this case, $\lambda(c)$ is viewed as the claim vector rather than c itself. The second regime occurs when the estate is sufficient to cover $\lambda(c)$ and in this case $\lambda(c)$ is considered as a right vector and $c - \lambda(c)$ is considered as the vector of claims in the remaining problem. Subsequently, within the first regime, the constrained equal award rule is used and within the second regime, the constrained equal loss rule is used. The resulting family of rules is called the claim-and-right family and is formally defined as follows.

Definition 7.3.2 *Let* $\lambda \in \Lambda$ *be a claim-and-right function. The* claim-and-right rule CR^λ *is defined by*

$$CR^\lambda(E,c) = \begin{cases} CEA(E,\lambda(c)) & \text{if } \sum_{i \in N} \lambda_i(c) \geq E, \\ \lambda(c) + CEL(E - \sum_{i \in N} \lambda_i(c), c - \lambda(c)) & \text{if } \sum_{i \in N} \lambda_i(c) < E, \end{cases}$$

for all $(E, c) \in BR^N$.

Using the four examples of claim-and-right functions discussed above, we have the following result.

Theorem 7.3.3 *CEA, CEL, AM and σ are claim-and-right rules.*

As a final result, we have that the claim-and-right family CR of rules coincides with the Increasing-Constant-Increasing family ICI of rules introduced by Thomson (2008). Note that the TAL-family (cf. Moreno-Ternero and Villar (2006)) is a part of the ICI-family, and as a result of Theorem 7.3.5, also a part of the CR-family. The ICI-family is defined as follows.

Definition 7.3.4 *(cf. Thomson (2008))*
Define a list $H := \{(F_k, G_k)\}_{k=1}^{n-1}$ *by a pair of functions from* $\mathcal{C}$ *to* $\mathbb{R}$ *such that for each* $c \in \mathcal{C}$ *the following holds:*

$$0 \leq F_1(c) \leq F_2(c) \leq ... \leq F_{n-1}(c) \leq G_{n-1}(c) \leq G_{n-2}(c) \leq ... \leq G_1(c) \leq \sum_{\ell \in N} c_\ell \quad (7.5)$$

$$G_k(c) = F_k(c) + \sum_{\ell=k+1}^{n} c_\ell - (n-k)c_k \qquad \text{for all } k \in \{1, ..., n-1\} \quad (7.6)$$

Denote by $\mathcal{H}^N$ *the family of lists and define for notational convenience* $F_0(c) = 0$, $G_n(c) = F_n(c) = F_{n-1}(c)$ *and* $G_0(c) = \sum_{\ell=1}^{n} c_\ell$. *An* ICI rule associated with $H = \{(F_k, G_k)\}_{k=1}^{n-1} \in \mathcal{H}^N$, *which we call* ICI^H, *is formulated for every* $(E, c) \in BR^N$ *as follows. When the estate increases from* $F_{i-1}(c)$ *to* $F_i(c)$, *each increment is divided equally over players* $\{i, ..., n\}$. *Furthermore, when the estate increases from* G_i *to* G_{i-1}, *each increment is divided equally over players* $\{i, ..., n\}$. *This gives the following formula:*

$$ICI_i^H(E, c) = \begin{cases} \sum_{\ell=1}^{i} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell} & \text{if } E \leq F_{n-1}(c) \text{ and } i < r, \\ \sum_{\ell=1}^{r-1} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell} + \frac{E - F_{r-1}(c)}{n+1-r} & \text{if } E \leq F_{n-1}(c) \text{ and } i \geq r, \\ \sum_{\ell=1}^{i} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell} & \text{if } E > F_{n-1}(c) \text{ and } i < r, \\ \sum_{\ell=1}^{i} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell} + \sum_{\ell=r+1}^{i} \frac{G_{\ell-1}(c) - G_\ell(c)}{n+1-\ell} + \frac{E - G_r(c)}{n+1-r} & \text{if } E > F_{n-1}(c) \text{ and } i \geq r, \end{cases}$$

in which

$$r := r(E,c) = \begin{cases} \min\{\ell \in \{1,...,n-1\} \mid E \leq F_\ell(c)\} & \text{if } E \leq F_{n-1}(c), \\ \min\{\ell \in \{1,...,n\} \mid E \geq G_\ell(c)\} & \text{if } E > F_{n-1}(c). \end{cases}$$

The following figure shows the main idea behind the proof of Theorem 7.3.5.

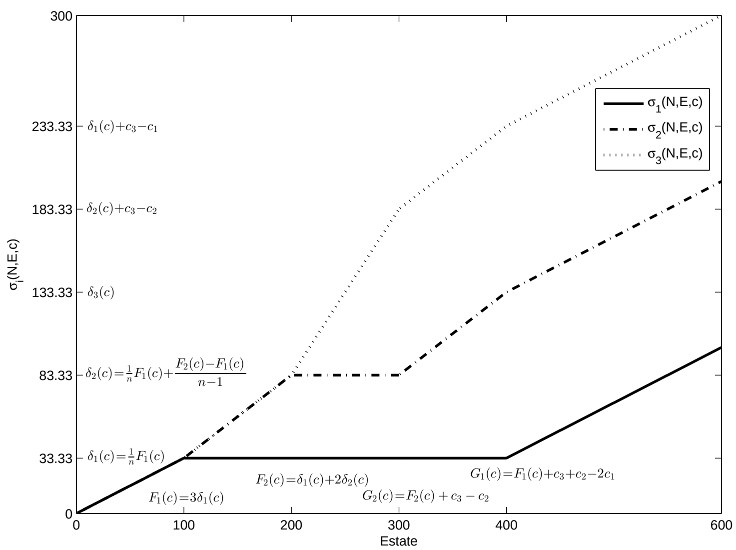

Figure 7.4: Path of $\sigma(\{1,2,3\},E,(100,200,300))$.

The ICI-family has as point of view the estate, i.e., in Figure 7.4, the cut-off points are when a certain estate has been reached, whereas the CR-family has the allocation of players as cut-off points.

Theorem 7.3.5 *Let f be a rule. Then, $f \in ICI$ if and only if $f \in CR$.*

Verve **Proof:** "only if part". Let $H = \{(F,G)\}_{k=1}^{n-1} \in \mathcal{H}^N$ be a list. Define, for each $c \in \mathcal{C}$, $\lambda^H(c) \in \mathbb{R}^N$ by $\lambda_i^H(c) = \sum_{\ell=1}^{i} \frac{F_\ell(c)-F_{\ell-1}(c)}{n+1-\ell}$ for all $i \in \{1,...,n\}$. We first prove that $\lambda^H \in \Lambda$.
For $i \in \{1,...,n-1\}$,

$$\lambda_i^H(c) = \sum_{\ell=1}^{i} \frac{F_\ell(c)-F_{\ell-1}(c)}{n+1-\ell}$$

$$
\begin{aligned}
&\geq \sum_{\ell=1}^{i-1} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell} \\
&= \lambda_{i-1}^H(c),
\end{aligned}
$$

in which the inequality uses that the sequence $F(c)$ weakly increases, *i.e.*, (7.5). Furthermore,

$$
\begin{aligned}
c_i - \lambda_i^H(c) &= c_i - \Big(\frac{F_1(c)}{n} + \sum_{\ell=2}^{i} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell}\Big) \\
&= c_{i-1} + c_i - c_{i-1} - \frac{F_i(c) - F_{i-1}(c)}{n+1-i} - \Big(\frac{F_1(c)}{n} + \sum_{\ell=2}^{i-1} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell}\Big) \\
&\geq c_{i-1} - \Big(\frac{F_1(c)}{n} + \sum_{\ell=2}^{i-1} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell}\Big) \\
&= c_{i-1} - \lambda_{i-1}^H(c).
\end{aligned}
$$

To clarify, at the inequality it is used that

$$
\begin{aligned}
F_i(c) - F_{i-1}(c) &= G_i(c) - \sum_{\ell=i+1}^{n} c_\ell + (n-i)c_i - \Big(G_{i-1}(c) - \sum_{\ell=i}^{n} c_\ell + (n+1-i)c_{i-1}\Big) \\
&= G_i(c) - G_{i-1}(c) + (n+1-i)(c_i - c_{i-1}) \\
&\leq (n+1-i)(c_i - c_{i-1}),
\end{aligned}
$$

in which the first equality uses (7.6) and the inequality uses that the sequence $G(c)$ is weakly decreasing, *i.e.*, (7.5). Note that the choice of $\lambda_n^H(c)$ implies both inequalities for $i = n$. Furthermore, $\lambda_1^H(c) = \frac{F_1(c)}{n} \geq 0$ and since

$$
F_1(c) = G_1(c) + (n-1)c_1 - \sum_{\ell=2}^{n} c_\ell \leq \sum_{\ell \in N} c_\ell + (n-1)c_1 - \sum_{\ell=2}^{n} c_\ell = nc_1,
$$

we have that $\lambda_1^H(c) = \frac{F_1(c)}{n} \leq c_1$. Hence, $\lambda^H \in \Lambda$.

Next, we prove that $ICI^H(E,c) = CR^{\lambda^H}(E,c)$. First, note that ICI^H divides the estate similar as in the CEA rule when $E \leq F_{n-1}(c)$, in other words, $ICI^H(E,c) = CEA(N,E,c^H)$, where $c_i^H = \lambda_i^H(c) = \sum_{\ell=1}^{i} \frac{F_\ell(c) - F_{\ell-1}(c)}{n+1-\ell}$ for all $i \in N$. Furthermore, the part of the estate which is larger than $F_{n-1}(c)$ is divided similar as in the CEL rule, in other words, $ICI^H(E,c) = c^H + CEL(N, E - F_{n-1}(c), c - c^H)$. Hence,

$$
ICI^H(E,c) = \begin{cases} CEA(N,E,c^H) & \text{if } E \leq F_{n-1}(c), \\ c^H + CEL(N, E - F_{n-1}(c), c - c^H) & \text{if } E > F_{n-1}(c), \end{cases}
$$

$$
\begin{aligned}
&= \begin{cases} CEA(N, E, \lambda^H(c)) & \text{if } \sum_{i \in N} \lambda_i^H(c) \geq E, \\ \lambda^H(c) + CEL(N, E - \sum_{i \in N} \lambda_i^H(c), c - \lambda^H(c)) & \text{if } \sum_{i \in N} \lambda_i^H(c) < E, \end{cases} \\
&= CR^{\lambda^H}(E, c),
\end{aligned}
$$

where it is used that $F_{n-1}(c) = F_n(c) = \sum_{i \in N} c^H = \sum_{i \in N} \lambda_i^H(c)$.

"if part". Let $\lambda \in \Lambda$. Define, for each $c \in \mathcal{C}$, $H^\lambda = \{(F_k^\lambda, G_k^\lambda)\}_{k=1}^{n-1}$ by

$$
\begin{aligned}
F_i^\lambda(c) &= \sum_{\ell=1}^{i-1} \lambda_\ell(c) + (n+1-i)\lambda_i(c) \\
G_i^\lambda(c) &= F_i^\lambda(c) + \sum_{\ell=k+1}^{n} c_\ell - (n-i)c_i,
\end{aligned}
$$

for all $i \in \{1, ..., n-1\}$. We will first prove that $H^\lambda = \{(F_k^\lambda, G_k^\lambda)\}_{k=1}^{n-1} \in \mathcal{H}^N$.

Clearly, by the definition, $H^\lambda = \{(F_k^\lambda, G_k^\lambda)\}_{k=1}^{n-1}$ satisfies condition (7.6) of a list. What is left to show is condition (7.5). Let $k \in \{2, ..., n-1\}$. Then,

$$
\begin{aligned}
F_k^\lambda(c) &= \sum_{\ell=1}^{k-1} \lambda_\ell(c) + (n+1-k)\lambda_k(c) \\
&\geq \sum_{\ell=1}^{k-1} \lambda_\ell(c) + (n+1-k)\lambda_{k-1}(c) \\
&= \sum_{\ell=1}^{k-2} \lambda_\ell(c) + (n+2-k)\lambda_{k-1}(c) \\
&= F_{k-1}^\lambda(c).
\end{aligned}
$$

The inequality uses that $\lambda_1(c) \leq ... \leq \lambda_n(c)$ (Definition 7.3.1). Furthermore,

$$
\begin{aligned}
G_k^\lambda(c) &= \sum_{\ell=1}^{k-1} \lambda_\ell(c) + (n+1-k)\lambda_k(c) - (n-k)c_k + \sum_{\ell=k+1}^{n} c_\ell \\
&= (n+1-k)(\lambda_k(c) - c_k) + \lambda_{k-1}(c) + \sum_{\ell=1}^{k-2} \lambda_\ell(c) + \sum_{\ell=k}^{n} c_\ell \\
&\leq (n+1-k)(\lambda_{k-1}(c) - c_{k-1}) + \lambda_{k-1}(c) + \sum_{\ell=1}^{k-2} \lambda_\ell(c) + \sum_{\ell=k}^{n} c_\ell
\end{aligned}
$$

$$
\begin{aligned}
&= (n+2-k)\lambda_{k-1}(c) + \sum_{\ell=1}^{k-2} \lambda_\ell(c) - (n+1-k)c_{k-1} + \sum_{\ell=k}^{n} c_\ell \\
&= G^\lambda_{k-1}(c).
\end{aligned}
$$

This time, the inequality uses that $c_1 - \lambda_1(c) \leq ... \leq c_n - \lambda_n(c)$ (Definition 7.3.1). Furthermore, $F^\lambda_1(c) = n\lambda_1(c) \geq 0$ and

$$
\begin{aligned}
G^\lambda_1(c) &= n\lambda_1(c) - (n-1)c_1 + \sum_{\ell=2}^{n} c_\ell \\
&\leq nc_1 - (n-1)c_1 + \sum_{\ell=2}^{n} c_\ell \\
&= \sum_{\ell\in N} c_\ell .
\end{aligned}
$$

Hence, the list $H^\lambda = \{(F^\lambda_k, G^\lambda_k)\}_{k=1}^{n-1} \in \mathcal{H}^N$. What is left to prove is that $ICI^{H^\lambda}(E,c) = CR^\lambda(E,c)$, in other words, show that $\lambda = \lambda^{H^\lambda}$. This follows from the same argument as used in the "only if part" together with the following derivation

$$
\begin{aligned}
\lambda_i^{H^\lambda}(c) &= \sum_{\ell=1}^{i} \frac{F^\lambda_\ell(c) - F^\lambda_{\ell-1}(c)}{n+1-\ell} \\
&\sum_{\ell=1}^{i} \frac{\sum_{k=1}^{\ell-1} \lambda_k(c) + (n+1-\ell)\lambda_\ell(c) - (\sum_{k=1}^{\ell-2} \lambda_k(c) + (n+2-\ell)\lambda_{\ell-1}(c))}{n+1-\ell} \\
&= \sum_{\ell=1}^{i} \frac{(n+1-\ell)(\lambda_\ell(c) - \lambda_{\ell-1}(c))}{n+1-\ell} \\
&= \lambda_i(c) - \lambda_0(c) \\
&= \lambda_i(c),
\end{aligned}
$$

where, for notational convenience, we define $\lambda_0(c) = 0$ and set
$F^\lambda_0(c) = (n+2-0)\lambda_0(c)$, which satisfies $F^\lambda_0(c) = 0$. □

Chapter 8

The proportionate nucleolus

This chapter, which is based on the preliminary manuscript Huijink, Borm, Hendrickx, and Reijnierse (2016), introduces the idea behind the proportionate nucleolus, which is a new type of nucleolus inspired by Faigle et al. (1998). Intuitively, the proportionate nucleolus selects the allocation that lexicographically maximizes the minimal ratio between the joint payoff to all players in a coalition and the worth of a coalition. This idea, however, creates problems, in particular when a coalition has worth zero. This chapter starts by illustrating these problems using examples and then formalizes the proportionate nucleolus. At the end of this chapter, we mention some preliminary results.

An obvious initial idea to define the proportionate nucleolus is by following the lines set out by the nucleolus, but using the proportionate excess of a coalition which is defined by $\frac{v(S)-\sum_{i\in S} x_i}{v(S)}$ for an allocation x. However, such a definition would have several drawbacks, which are illustrated in the following examples.

Example 8.1.1 Consider the two person game $v \in TU^N$ with $N = \{1, 2\}$ given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$v(S)$	-2	-1	1

Lexicographically minimizing the proportionate excesses among all efficient allocation will lead to the allocation $x = (\frac{2}{3}, \frac{1}{3})$ since the corresponding proportionate excess vector is $(1\frac{1}{3}, 1\frac{1}{3}, 0)$ with corresponding vector of coalitions $(\{1\}, \{2\}, N)$. However, this implies that player 2 receives less than player 1, while player 2 has a higher individual worth than player 1. $\triangleleft$

Example 8.1.1 shows that proportionality and negative worths do not go together. This problem can be avoided by restricting the domain of the proportionate nucleolus to non-negative games. Another important issue, however, is that the excesses are not defined for coalitions with worth zero. The following example shows that we should not just ignore the coalitions with worth zero.

Example 8.1.2 Consider the two person game $v \in TU^N$ with $N = \{1, 2\}$ given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$v(S)$	0	1	1

When ignoring coalition $\{1\}$, there is no efficient allocation in $\mathbb{R}^N$ which lexicographically minimizes the accompanying proportionate excess vector. The allocation $(-A, A+1)$ for any $A \in \mathbb{R}_+$ has as proportionate excess vector $(0, -A)$ for the vector of coalitions $(N, \{2\})$. ◁

The issue in the previous example can be avoided by requiring the outcome to be an efficient allocation and non-negative. However, even so, ignoring the coalitions with worth zero will bite unicity.

Example 8.1.3 Consider the two person game $v \in TU^N$ with $N = \{1, 2\}$ given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$v(S)$	0	0	1

Any vector in $conv\{(1, 0), (0, 1)\}$ will lexicographically minimize the maximal excess when ignoring all coalitions with worth zero. Obviously, for this game, the desired efficient outcome is $(\frac{1}{2}, \frac{1}{2})$ on the basis of symmetry. ◁

The following example shows that proportionality implies that the solution concept will not satisfy so called covariance.

Example 8.1.4 Consider the two person game $v \in TU^N$ with $N = \{1, 2\}$ given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$v(S)$	1	2	6

The vector $(2,4)$ lexicographically minimizes the maximal excess. However, if we modify the game v by adding the additive game which assigns $(1,1)$ to the respective players, we obtain the following game $w \in TU^N$:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$w(S)$	2	3	8

This vector which lexicographically minimizes the maximal excess is $(3\frac{1}{5}, 4\frac{4}{5})$, which is unequal to $(2,4)+(1,1)=(3,5)$. $\triangleleft$

A way out could be to restrict the domain of games under investigation even further to strictly positive games. Instead, we propose another route to find a unique proportionate nucleolus for each non-negative game. We are going to modify the initial idea of proportional excesses in the following way.

Definition 8.1.6 *Let $v \in TU_+^N$ be a non-negative game ($v(S) \geq 0$ for all $S \in 2^N$), let $\varepsilon > 0$ and let $x \in E^N(v)$ where $E^N(v) = \{y \in \mathbb{R}^N \mid \sum_{i \in N} y_i = v(N)\}$. Then, the ε-*proportionate ratio *of any non-empty coalition $S \in 2^N \backslash \{\varnothing\}$ is defined by*

$$Q^\varepsilon(v,S,x) = \frac{\sum_{i \in S} x_i}{v(S)+\varepsilon}.$$

The *ε-proportionate ratio vector* of any game $v \in TU_+^N$ with $N = \{1,...,n\}$ is defined by $\theta^\varepsilon(v,x) \in \mathbb{R}^{2^n-1}$ and has as its coordinates the ratios of all non-empty coalitions arranged in a weakly increasing order. In other words, $\theta_k^\varepsilon(v,x) \leq \theta_{k+1}^\varepsilon(v,x)$ for all $k \in \{1,...,2^n-2\}$. We want to maximize the ε-proportionate ratio vector lexicographically.

Definition 8.1.7 *Let $v \in TU_+^N$ and $\varepsilon > 0$. Then, the set $\mathcal{X}^\varepsilon(v)$ is given by*

$$\mathcal{X}^\varepsilon(v) = \{x^\varepsilon(v) \in E^N(v) \mid \theta^\varepsilon(v, x^\varepsilon(v)) \geq_L \theta^\varepsilon(v,y) \text{ for all } y \in E^N(v)\}.$$

Note that $\mathcal{X}^\varepsilon(v)$ is a "general nucleolus" (Maschler et al. (1992)), which implies that this set contains exactly one point. For completeness, we present another proof.

Lemma 8.1.8 *Let $v \in TU_+^N$ and let $\varepsilon > 0$. Then, $|\mathcal{X}^\varepsilon(v)| = 1$.*

Proof: First, we prove that the set $\mathcal{X}^\varepsilon(v)$ is non-empty. Then, we prove that the set consist of one point only.
Non-emptiness: Note that the set $\{x(v) \in E^N(v) \mid \theta^\varepsilon(v, x(v)) \geq_L \theta^\varepsilon(v, \frac{v(N)}{n}e^N)\}$ is compact. Since $x^\varepsilon(v) \subset \{x(v) \in E^N(v) \mid \theta^\varepsilon(v, x(v)) \geq_L \theta^\varepsilon(v, \frac{v(N)}{n}e^N)\}$ and $\theta^\varepsilon(\cdot)$ is a continuous function on $E^N(v)$, there exists a lexicographic maximum on this set. Hence, $\mathcal{X}^\varepsilon(v)$ is not empty.

Unicity: Let $x, y \in \mathcal{X}^\varepsilon(v)$ with $x \neq y$. Clearly, $\theta^\varepsilon(v,x) = \theta^\varepsilon(v,y)$. Consider $z = \frac{1}{2}(x+y) \in E^N(v)$, for which we show that $\theta^\varepsilon(v,z) > \theta^\varepsilon(v,x)$ which contradicts the fact that $x \in \mathcal{X}^\varepsilon(v)$. Let

$$\mathcal{S} = \{S \in 2^N \backslash \{\varnothing, N\} \mid \sum_{i \in S} x_i \neq \sum_{i \in S} y_i\},$$
$$\mathcal{S}^* = \{S \in \mathcal{S} \mid Q^\varepsilon(v,S,x) \leq Q^\varepsilon(v,T,x) \text{ for all } T \in \mathcal{S}\}.$$

Note that $\mathcal{S}$ and $\mathcal{S}^*$ are non-empty.
Let $Q = Q^\varepsilon(v, S^*, x)$ for any $S^* \in \mathcal{S}^*$ and let $S \in 2^N \backslash \{\varnothing, N\}$. The proof is divided into cases, depending on the ratio of the coalition S.

Case 1: Suppose $Q^\varepsilon(v,S,x) < Q$. Then, by definition of $\mathcal{S}$ and $\mathcal{S}^*$, we have that $\sum_{i \in S} x_i = \sum_{i \in S} y_i = \sum_{i \in S} z_i$, which implies that
$Q^\varepsilon(v,S,z) = Q^\varepsilon(v,S,x) = Q^\varepsilon(v,S,y)$.

Case 2: Suppose $Q^\varepsilon(v,S,x) = Q$ and $S \notin \mathcal{S}^*$. Again, by definition of $\mathcal{S}$ and $\mathcal{S}^*$, we have that $\sum_{i \in S} x_i = \sum_{i \in S} y_i = \sum_{i \in S} z_i$, which implies that
$Q^\varepsilon(v,S,z) = Q^\varepsilon(v,S,x) = Q^\varepsilon(v,S,y)$.

Case 3: Suppose $Q^\varepsilon(v,S,x) = Q$ and $S \in \mathcal{S}^*$. Using case 1, we can assume that $Q^\varepsilon(v,S,y) \geq Q$. Furthermore, since $S \in \mathcal{S}^*$, we have that $\sum_{i \in S} x_i \neq \sum_{i \in S} y_i$. This implies that $Q^\varepsilon(v,S,y) > Q$, which implies that
$Q^\varepsilon(v,S,z) = \frac{\sum_{i \in S} \frac{1}{2}(x_i+y_i)}{v(S)} = \frac{1}{2}Q^\varepsilon(v,S,x) + \frac{1}{2}Q^\varepsilon(v,S,y) > Q$.

Case 4: Suppose $Q^\varepsilon(v,S,x) > Q$. Again, using case 1, we can assume that $Q^\varepsilon(v,S,y) \geq Q$. Hence, $Q^\varepsilon(v,S,z) > Q$.

Since $Q^\varepsilon(v,S,z) \geq Q^\varepsilon(v,S,x)$ for all S such that $Q^\varepsilon(v,S,x) \leq Q$, and since $\mathcal{S}^*$ is non-empty, there exists at least one coalition S such that

$Q^\varepsilon(v,S,z) > Q^\varepsilon(v,S,x) = Q$, we have that $\theta^\varepsilon(v,z) > \theta^\varepsilon(v,x)$. □

Definition 8.1.9 *Let $v \in TU_+^N$ and let $\varepsilon > 0$. Then, $x^\varepsilon(v)$ is defined as the unique element in $\mathcal{X}^\varepsilon(v)$.*

The allocation $x^\varepsilon(v)$ is the unique allocation, in other words a singleton, which lexicographically maximizes the ratios. Moreover, it also satisfies the Kohlberg criteria discussed in Section 6.3. We use the following variant of Proposition 6.3.3.

Proposition 8.1.10 *Let $v \in TU_+^N$, let $\varepsilon > 0$ be such that $Core(v) \neq \varnothing$, and let $x \in E^N(v)$. Then, $x = x^\varepsilon(v)$ if and only if there exists a sequence $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_\tau$ of non-empty subcollections of $2^N \backslash \{\varnothing, N\}$ such that $\mathcal{D}_r \subseteq \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ for all $r \in \{1,...,\tau\}$ that satisfy the following properties:*

(A) for all $r \in \{1,...,\tau\}$ the collection $\bar{\mathcal{D}}_r = \bigcup_{k=1}^r \mathcal{D}_k$ is balanced and $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$.

(B) for all $r \in \{1,...,\tau\}$ and all $T \in \mathcal{D}_r$ it holds that $Q^\varepsilon(v,T,x) = \min\limits_{S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})} Q^\varepsilon(v,S,x)$.

The proof is a replication of the proof of Proposition 6.3.3 with minor changes and is therefore omitted. Subsequently, the proportionate nucleolus can be defined as the limit of the sequence x^ε when ε tends to 0. The following example illustrates the proportionate nucleolus for the games in Examples 8.1.2 and 8.1.3.

Example 8.1.5 (Examples 8.1.2 and 8.1.3 revisited.)
Consider the two person game of Example 8.1.2 with $N = \{1,2\}$, $v(\{1\}) = 0$, $v(\{2\}) = 1$ and $v(N) = 1$. Then, $x^\varepsilon(v) = (\frac{\varepsilon}{1+2\varepsilon}, \frac{1+\varepsilon}{1+2\varepsilon})$ since the corresponding ratios are given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$v(S)$	0	1	1
$Q^\varepsilon(v,S,x^\varepsilon(v))$	$\frac{1}{1+2\varepsilon}$	$\frac{1}{1+2\varepsilon}$	$\frac{1}{1+\varepsilon}$

and, with $\tau = 1$ and $\mathcal{D}_1 = \{1,2\}$, all conditions of Proposition 8.1.10 are satisfied. Furthermore, $propnucl(v) = \lim_{\varepsilon \downarrow 0} x^\varepsilon(v) = (0,1)$.

For the game of Example 8.1.3, with $N = \{1,2\}$, $v(\{1\}) = 0$, $v(\{2\}) = 0$ and $v(N) = 1$, we have $x^\varepsilon(v) = (\frac{1}{2}, \frac{1}{2})$, since the corresponding ratios are given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$v(S)$	0	0	1
$Q^\varepsilon(v, S, x^\varepsilon(v))$	$\frac{1}{2\varepsilon}$	$\frac{1}{2\varepsilon}$	$\frac{1}{1+\varepsilon}$

and, with $\tau = 1$ and $\mathcal{D}_1 = \{1,2\}$, all conditions of Proposition 8.1.10 are satisfied. Furthermore, $propnucl(v) = \lim_{\varepsilon \downarrow 0} x^\varepsilon(v) = (\frac{1}{2}, \frac{1}{2})$. $\triangleleft$

Huijink, Borm, Hendrickx, and Reijnierse (2016) provide a direct Kohlberg (Kohlberg (1971)) type of characterization for the proportionate nucleolus which does not rely on the Kohlberg criteria for the ε-proportionate nucleolus $x^\varepsilon(v)$. Moreover, it defines a suitable reduced game property that can be used to characterize the proportionate nucleolus.

Example 8.1.6 (Example 1.1.2 revisited.)
Consider the problem of the two companies in Example 1.1.2. Company X (player 1) has costs 105, company Y (player 2) has costs 72. When cooperating, the companies have costs 139. Note that in the corresponding game we consider costs rather than rewards. For a TU-game v, the proportionate nucleolus maximizes the minimal ratio. Translating the ideas behind the proportionate nucleolus to a cost setting, we find that the proportionate nucleolus of a cost game minimizes the maximal ratio. The problem gives the following two person cost game c, with $N = \{1,2\}$, $c(\{1\}) = 105$, $c(\{2\}) = 72$ and $c(N) = 139$. Then, $x^\varepsilon(c) = (\frac{(105+\varepsilon)\cdot 139}{177+2\varepsilon}, \frac{(72+\varepsilon)\cdot 139}{177+2\varepsilon})$ since the corresponding ratios are given by:

S	$\{1\}$	$\{2\}$	$\{1,2\}$
$Q^\varepsilon(c, S, x^\varepsilon(c))$	$\frac{139}{177+2\varepsilon}$	$\frac{139}{177+2\varepsilon}$	$\frac{139}{139+\varepsilon}$

and, decreasing the costs (and ratio) of one player will increase the costs (and ratio) of the other player. Hence, $propnucl(c) = \lim_{\varepsilon \downarrow 0} x^\varepsilon(c) = (82\frac{27}{59}, 56\frac{32}{59})$. $\triangleleft$

Appendix A

Details of chapter 2

The four confidential interviews were conducted using a questionnaire as a starting point. The interviewees received the questions beforehand and the interview took place in a face-to-face meeting. During these face-to-face meeting, follow up questions were asked. In one case, the interviewee answered the questions by email and the follow up questions were asked via the telephone. The questions (in Dutch) can be found in Section A.2. For convenience, an excerpt of the opportunities and impediments is translated in English and can be found in the following section.

A.1 Excerpt of the interviews

Opportunities:

- Without cooperation we were not able to guarantee 24 hour delivery. The cooperation increased our service level which improved our market position.
- Due to the cooperation we are able to reduce our costs, while we have more business.

Selection impediments:

- The reason that we chose the pricing based structure is that we do not want to share all our information and we want to make our own decisions.
- For a cooperation to work, it is needed that the companies need each other everyday and that the costs can be reduced when cooperating.
- The companies in the cooperation were chosen due to their geographical focus. Sometimes, additionally for their type of clients/orders and the resources.

- If the companies do not trust each other, the cooperation will fall apart. The energy which is needed to make the cooperation a success will be missing.
- The main risks of cooperation are the dependency on the others and that a company stops due to, for example, bankruptcy or retirement.
- A previous cooperation failed due to opportunistic behavior of a company.
- It is necessary to make sure that the company who delivers an outsourced order does not steal the client.

Operational impediments:

- It is important that each company has gains everyday from cooperating. The gains are divided by the pricing mechanism.
- Sometimes it is difficult to determine whether an order should be outsourced or delivered with the own fleet. There are no good software packages available.
- For a good cooperation, it is necessary that the companies can share the delivery information quickly and easily using a good transportation management system.
- In order to have a good division of the gains, it is necessary that there is a balance within the cooperation. Companies need to outsource similar amounts to each partner and the outflow and inflow should be similar.
- There should be no discrimination between orders of own clients and orders from the partners.

A.2 Questionnaire (in Dutch)

Vragen over het samenwerkingsverband

- Hoe is het samenwerkingsverband ontstaan?
- Hoe werkt het samenwerkingsverband?
- Hoe is het georganiseerd?
- Wie leidt het verband?

- Wat is gecentraliseerd en wat gedecentraliseerd?
- Wordt het van bovenaf aangestuurd of bepalen de partners zelf welke pakketjes ze bezorgen en uitwisselen?
- Hoe werkt het uitwisselen van de orders? (Via een hub en spoke met een centraal depot?)
- Van wie is het depot?
- Hoe worden de kosten van het depot verdeeld?
- Of is er een web?
- Hoe worden de kosten van het web verdeeld?
- Hoe wordt het web bepaald?
- Vervoert het samenwerkingsverband zelf ook orders?
- Zo ja, verschilt dit vervoer met het vervoer van de partners?
- Welke afspraken zijn er gemaakt en worden de afspraken nageleefd?
- Heeft elk bedrijf een vooraf afgesproken jacht- of bestel gebied?
- Wordt er centraal winst gemaakt?
- Hoe wordt die winst verdeeld?
- Wat zijn de centrale kosten?
- Hoe worden de kosten verdeeld?
- Wat te doen bij discussies?
- Wat zijn de voordelen van het samenwerken?
- Maakt elk bedrijf elke dag winst door samen te werken?
- Wat zijn de nadelen aan het samenwerken?
- Waarom werkt dit samenwerken wel?

- Stel: we hebben een afspraak niet, werkt het dan wel?
- Zijn er ook samenwerkingsverbanden/afspraken die niet werken?
- Zijn er risico's?
- Hoe zijn deze verdeeld?
- Wat zijn de plannen voor de toekomst met het samenwerkingsverband?
- Uitbreiding door meer leden?

Partner Selectie

- Hoe zijn de partners gevonden?
- Moesten de partners voldoen aan bepaalde eisen?
- Bv. grootte, solvabiliteit, geografische plaatsing, klantenbestand, orderkarakteristieken, soort trucks?

Operationele vragen voor de vervoerders

- Hoe bepalen de bedrijven welke orders uitgewisseld moeten worden?
- Welke regels gebruiken ze?
- Hoe zien die regels eruit?
- Hoeveel procent wordt bepaald met deze regels?
- Hoe wordt de beslissing over de andere pakketjes gemaakt?
- Wordt er ook soms gekeken of een pakketje niet beter zelf gedaan kan worden?
- i.e., een pakketje dat volgens de regel uitgewisseld zou worden toch zelf bezorgen.
- Zijn de bedrijven verplicht tot uitwisselen?
- i.e., mogen ze alleen in hun eigen gebied bestellen?
- Moeten ze de orders die de andere partners uitwisselen aannemen?

- Zijn er afspraken waar deze pakketjes aan moeten voldoen?
- Wat zijn de kosten van het uitwisselen van een order?
- Wat moet een bedrijf betalen en wat krijgt het als het een order van een partner ontvangt om te bezorgen?
- Hoe worden deze prijzen vastgesteld?
- Wanneer en hoe wordt er gepland?
- Wordt er eerst bepaald wat wordt uitgewisseld en worden de routes pas gepland als de uitgewisselde orders binnen zijn?
- Wordt het met de hand gepland of met software, welke software?
- Worden er ook pick-up and delivery orders gedaan door de bedrijven?
- Of gaat alles via het depot van het bedrijf?
- Wat is de verdeling tussen pick-up and delivery en pakjes die via het depot gaan?
- Hoeveel orders bezorgen jullie gemiddeld per dag?
- Hoeveel worden er uitgewisseld naar de partners?
- Hoeveel worden er ontvangen?
- En totaal in het hele netwerk per dag?
- Bezorgt en uitgewisseld?
- Wat zijn de specificaties van de gemiddelde order?
- Grootte, volume, time-window etc.
- Hoe ver ligt het ophaal en aflever adres uit elkaar?
- Of hoeveel van de orders liggen buiten het eigen gebied?
- Hoeveel stops maakt een truck gemiddeld?
- Wat is de gemiddelde stop afstand?

- Zijn er ook routes die altijd worden gereden?
- i.e. vaste routes.
- Wat zijn de plannen voor de toekomst van het bedrijf?

Out-insourcen aan derden

- Gebruiken jullie derden om pakjes te bestellen?
- Wordt alleen een truck met chauffeur gehuurd of gaat de order naar een ander bedrijf?
- Ook in de stad, last mile?
- Ook voor pick-up en delivery?
- Leveren jullie ook pakjes van derden?
- Idem as hierboven.
- Leveren jullie een truck en chauffeur of gaat de order door het systeem?

Stadsdistributie

- Waarin verschilt stadsdistributie met “normale” vervoersopdrachten?
- Hebben deze orders andere karakteristieken?
- Grootte, volume, time-window etc.
- Heeft het bezorgen in de stad andere eigenschappen dan de rest?
- Is het tijdsvenster de boosdoener?
- Hoeveel winkelstraten kan een truck doen?
- Kan een truck verschillende steden doen?
- Hoe vol zit een truck voor een stad?
- Hoeveel stops doet een truck in de stad?
- Wat is de stopafstand in de stad?

- Hoeveel orders worden er in de stad bezorgd? (binnenstad en wijken, geen industrieterrein)
- In verhouding met het totale aantal.
- Wijs je een stad toe aan een enkele partner?

Punten die gemist zijn? Waar kan volgens jullie het beste onderzoek naar gedaan worden?

Appendix B

Details, parameters and test results of chapter 3

B.1 Details and parameters of heuristics

The remove-and-insert move from Archetti et al. (2007) removes randomly a random integer in $[LB, UB]$. The parameters for the two heuristics are presented in Table B.1.

Heuristic	Name	LB	UB
LNS-Fast	Small	3	8
	Large	5	12
LNS-Slow	Small	4	9
	Large	7	12

Table B.1: The parameters for the remove-and-insert move.

The cyclic move, which are base upon the moves of Stenger et al. (2013), draws a random integer in $[LB, UB]$. This number represents the number of routes which are involved in this cyclic move. The first driven route is selected based on one of the following four criteria.

1. *Random:* All driven routes have the same probability of being selected.
2. *Distance:* The probability is proportional to the total distance traveled in this route. Hence, longer routes have higher probability of being selected.
3. *Unit Distance:* Similar to *Distance*, but now the distance traveled divided by the demand used in the route.

4. *Distance to Others:* The probability is proportional to the inverse of the least insertion cost of an order in this route to other routes. In this way, a route is favored which has orders close to another route.

From this route, a sequence of orders with length a random integer in $[OrdersLB, OrdersUB]$, is selected based on one of the following criteria.

1. *Random:* All orders have the same probability of being selected.

2. *Distance:* The probability is proportional to the sum of the distance costs from the location before the sequence to the starting location of this sequence and the distance costs from the last location of this sequence to the location after the sequence. Hence, sequences that are far away from the other orders in the route have a higher probability.

3. *Average Insertion Distance:* The probability is inversely proportional to the average insertion distance of the sequence in the closest non-chosen driven route. By doing so, the orders that are close to another driven route have a higher chance of being chosen.

4. *Demand:* The probability is proportional to the total demand of the sequence.

These orders are inserted into a driven route which is not yet modified by this move and that has the least insertion costs of the sequence of orders. To facilitate a closed cycling, we additionally allow the first route to be chosen when selecting the final route. Again, a sequence of orders, with as length a random integer in $[SecondLB, SecondUB]$, is selected based on the same criteria, where the orders just inserted are not allowed to be selected.

Similarly, the second type of cyclic moves (CyclicCandidate) switches orders between driven routes and candidate routes. Again, a random integer in $[LB, UB]$ is drawn. The first candidate route is selected based on one of the following four criteria.

1. *Random:* All candidate routes have the same probability of being selected.

2. *Profit:* The probability is proportional to $1 + Profit$ if the Profit is positive and $\frac{-1}{-1+Profit}$ otherwise, where the Profit is the outsource costs of the sequence minus the distance costs within this sequence. Furthermore, since the virtual route, which contains all the undelivered orders that do not belong to candidate

routes, does not have a distance, it receives half the value of the candidate route with the lowest value. Hence, routes that have a smaller loss are more likely to be selected.

3. *Unit Profit:* The probability is proportional to $1+ProfitOrder$ if ProfitOrder is positive and $\frac{-1}{-1+ProfitOrder}$ otherwise, where ProfitOrder is the outsource costs of the best order minus the insertion costs of this order divided by its demand. Again, the virtual route has half the value of the lowest one.

4. *Unit Insertion Profit in Driven route:* Now the probability is proportional to $1+ProfitIns$ if ProfitIns is positive and $\frac{-1}{-1+ProfitIns}$ otherwise, where ProfitIns is the outsource costs minus insertion costs in the closest driven route divided by the demand of the best order. In this way, the route is favored which has the order which can be most profitable inserted into a driven route.

From this candidate route (or virtual route) a sequence of orders with length a random integer in $[OrdersLB, OrdersUB]$ is drawn based on the following criteria.

1. *Random:* All orders have the same probability of being selected.

2. *Unit Profit:* The probability is proportional to $1 + \frac{Profit}{Demand}$ if Profit is positive and $\frac{-1}{-1+Profit*Demand}$ otherwise, where Profit is the outsource costs minus the distance costs of the sequence and Demand is the demand of the order sequence. There is no sequence in the virtual route. Therefore, the first order is chosen with probability proportional to its outsource costs divided by the demand. Then, other orders are added at the beginning or the end of the sequence, where the probability is proportional to the price minus the least distance costs to the end or beginning divided by the demand of the order. By doing so, the orders that have the most unit profit are chosen to be removed.

3. *Insertion Profit:* The probability is proportional to $1+ProfitIns$ if ProfitIns is positive and $\frac{-1}{-1+ProfitIns}$ otherwise, where ProfitIns is the outsource costs of the sequence of orders minus the insertion costs in the not changed driven route closest for this sequence. Again, the virtual route needs a different treatment, namely, each order receives a probability proportional to its outsource costs minus the insertion costs and they are drawn until the needed number of orders is reached. In this way, the profitable sequences to insert into other routes are taken out of the candidate route.

4. *Unit Insertion Profit:* Similar to *Insertion Profit* but now divided by the demand of the orders. Hence, the unit wise profitable sequences are removed out of the candidate route.

The selected orders are inserted into the driven route which is not yet modified by this move and that has the least insertion costs of the sequence of orders. To facilitate a closed cycling, we additionally allow the first route to be chosen when selecting the final route. Again, a sequence of orders, with length a random integer in $[SecondLB, SecondUB]$, is selected but since the route is driven, we need different criteria.

1. *Random:* All orders have the same probability of being selected.

2. *Distance:* The probability is proportional to the sum of the distance costs from the location before the sequence to the starting location of this sequence and the distance costs from the last location of this sequence to the location after the sequence. Hence, sequences that are far away from the other orders in the route have a higher probability of being removed.

3. *Average Profit:* The probability is proportional to $\frac{1}{1+Profit}$ if Profit is positive and $1 - Profit$ otherwise, where Profit is the outsource costs of the sequence of orders minus the distance costs. By doing so, the orders which have the least profit have the highest probability to be removed.

4. *Unit Average Profit:* Similar to *Average Profit* but now divided by the demand of the orders.

The orders selected from a driven route are inserted into the next route, which could be driven or not, which is not yet chosen, does not have an average profit of more than two times the average profit of the removed orders and has the least insertion costs of the orders. Similar as in Stenger et al. (2013) we have an adaptive mechanism for the selection criteria.

For the LNS-Slow, we changed the Demand criterion to the following.

4. *Distance/Average Insertion Distance:* The probability is proportional to the sum of the distance costs from the location before the sequence to the starting location of this sequence and the distance costs from the last location of this sequence to the location after the sequence, divided by the average insertion

distance of the sequence in the closest not changed driven route. Hence, sequences that are not close to other orders in this route but are close to other routes are favored.

The parameters are presented in Table B.2.

Heuristic	name	LB	UB	$OrdersLB$	$OrdersUB$	$SecondLB$	$SecondUB$
LNS-Fast	Small, one-way	2	2	3	4	0	0
	Small	2	2	2	3	2	3
	Medium	2	2	3	4	3	4
	Large	2	4	2	4	2	4
	Candidate, small	2	2	2	3	2	3
	Candidate, medium	2	2	5	6	2	3
	Candidate, large	2	4	2	4	2	4
LNS-Slow	Small, one-way	2	2	3	4	0	0
	Small	2	2	2	4	2	4
	Medium	2	4	3	4	3	4
	Large	3	5	2	5	2	5
	Candidate, small	2	2	3	5	2	4
	Candidate, medium	2	3	2	4	2	4
	Candidate, large	3	5	2	5	2	5

Table B.2: The parameters for the cyclic move.

The create move selects an outsourced order which has the highest value, where the value is the outsource costs minus the distance costs of this order to the ten closest outsourced orders.

The shifting procedure assigns to each order which does not belong to a receiving route a score which is based on the insertion profit of the order. If the order is outsourced, then the value is $\frac{p_i - InsertionCosts_i}{q_i}$, where $InsertionCosts_i$ is the insertion costs in the new route. However, if the order is delivered the value is $\frac{p_i - InsertionCosts_i}{q_i}$ multiplied (divided) by $(1 + DistanceNow_i - InsertionCosts_i)$ when the $DistanceNow$, which is the detour costs to deliver the order in its current route, is larger (smaller) than the insertion costs. The order with the highest value and that feasible can be inserted, is chosen. The LNS-Fast has two create moves while the LNS-Slow has only one.

The destroy move randomly selects a driven route and all orders in this route are outsourced. The LNS-Fast has two destroy moves while the LNS-Slow has only one. For the LNS-Slow, the destroy move is followed, with a 50% probability, by the create move.

The split move randomly selects a driven route and the route is split where the distance between two orders is the largest. The shifting procedure is applied after the split move.

The bomb move randomly selects an order as center and repeatedly outsources the delivered order that has its location closest to the center. The move stops when either the ratio of the orders that the move outsources is Max, or when the number of routes which have been affected is $MaxRoutes$, or when the ratio of the orders that affected is at least Min together with the number of routes affected is at least $MinRoutes$. For the LNS-Slow, we draw Max randomly in $[MaxLB, MaxLB]$, $MaxRoutes$ in $[MaxRoutesLB, MaxRoutesUB]$, Min in $[MinLB, MinUB]$ and $MinRoutes$ in $[MinRoutesLB, MinRoutesUB]$. After the bomb move, the solution is repaired using the shifting procedure. The parameters for the bomb move are presented in Table B.3, where m is the number of routes available, and Table B.4.

Name	$Min(\%)$	$Max(\%)$	$MinRoutes$	$MaxRoutes$
Small	10	40	3	6
Large	12.5	40	3	m-1

Table B.3: The parameters for the bomb move for the LNS-Fast.

	Min (%)		Max (%)		MinRoutes		MaxRoutes	
Name	LB	UB	LB	UB	LB	UB	LB	UB
Small	10	20	30	40	2	4	3	5
Large	20	30	40	50	3	6	7	9

Table B.4: The parameters for the bomb move for the LNS-Slow.

The reseed move draws a random integer in $[LB, UB]$ of routes. From each route a sequence of orders, with length a random integer in $[OrderLB, OrderUB]$, stays in the route and the other orders are outsourced. Again, the solution is repaired using the shifting procedure. For the LNS-Slow, we draw a sequence of orders with as length a random integer in $[\#O \cdot OredrLB, \#O \cdot OrderUB]$, where $\#O$ is the number of orders in the route. The shifting procedure is used to repair the solution. The parameters for the reseed move are presented in Table B.5.

Heuristic	Name	LB	UB	$OrdersLB$	$OrdersUB$
LNS-Fast	Small	2	6	5	11
	Large	5	m	6	11
LNS-Slow	Small	1	3	30%	60%
	Large	4	6	60%	70%

Table B.5: The parameters for the reseed move.

Each of these moves if followed by a tabu search heuristic which has the following parameters. The minimal number of iterations is presented by Min, which is a random integer in $[MinLB, MinUP]$. Similarly, the maximum number of iterations is Max, the maximum number of iterations without improvement $NoImpr$, after how many iterations to recalculate the infeasibility penalty by $Infeas$, increase (decrease) the penalty by Up ($Down$), and the tabu duration is a random integer in $[TLower, TUpper]$.

	Min		Max		NoImpr		Up		Down		Infeas		TLower		TUpper	
Heuristic	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*
LNS-Fast	25	45	300	500	25	35	1.5	3	1.5	3	3	5	4	9	12	27
LNS-Slow	15	20	166	233	20	40	1.5	3	1.5	3	3	6	5	10	13	26

Table B.6: The parameters for the tabu search after the move.

The final move is a slightly longer tabu search heuristic. The parameters are presented in Table B.7.

	Min		Max		NoImpr		Up		Down		Infeas		TLower		TUpper	
Heuristic	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*	*LB*	*UB*
LNS-Fast	200	300	1000	1500	100	150	1.5	3	1.5	3	3	5	4	9	12	27
	200	300	1000	1500	100	150	1.5	3	1.5	3	3	5	4	9	12	27
LNS-Slow	150	200	500	700	30	60	1.5	3	1.5	3	3	6	5	10	13	26

Table B.7: The parameters for the tabu search move.

Finally, the minimal number of iterations for the LNS-Fast and LNS-Slow are Min, the maximum Max, and the heuristic stops after $NoImpr$ iterations without improvement of the local solution. The parameters are presented in Table B.8.

Heuristic	Min	Max	$NoImpr$
LNS-Fast	200	1400	40
LNS-Slow	150	750	34

Table B.8: The parameters for LNS.

B.2 Detailed test results

		RIP Bolduc et al. (2008)		TS25 Côté and Potvin (2009)			TS Potvin and Naud (2011)		TS+ Potvin and Naud (2011)	
Instance	BKS	single	CPU(s)	best	avg	CPU(s)	best	CPU(s)	best	CPU(s)
CE-01	1119.47	1132.91	25	1119.47	1119.47	8.5	1119.47	24.3	1119.47	24.9
CE-02	1814.52	1835.76	73	1814.52	1816.07	12.5	1814.52	33.0	1814.52	33.9
CE-03	1919.05	1959.65	107	1924.99	1930.28	34.7	1921.10	78.6	1930.66	81.0
CE-04	2505.39	2545.72	250	2515.50	2526.41	83.3	2525.17	193.2	2525.17	200.6
CE-05	**3081.42**	3172.22	474	3097.99	3112.25	128.3	3113.58	309.9	3117.10	353.2
CE-06	1207.47	1208.33	25	1207.47	1207.47	9.9	1207.47	25.5	1207.47	25.2
CE-07	2004.53	2006.52	71	2006.52	2010.96	14.0	2006.52	32.7	2006.52	34.0
CE-08	2052.05	2082.75	110	2055.64	2063.06	36.9	2060.17	85.1	2056.59	81.6
CE-09	**2418.64**	2443.94	260	2429.19	2433.86	83.3	2438.43	185.3	2435.97	188.2
CE-10	**3373.42**	3464.90	478	3393.41	3402.72	129.6	3406.82	311.1	3401.83	345.7
CE-11	2330.94	2333.03	195	2330.94	2336.59	54.6	2353.39	126.3	2332.36	131.0
CE-12	1952.86	1953.55	128	1952.86	1961.49	24.2	1952.86	60.4	1952.86	59.5
CE-13	2858.83	2864.21	188	2859.12	2863.96	53.7	2882.70	130.0	2860.89	132.1
CE-14	2213.02	2224.63	110	2214.14	2220.23	24.8	2219.97	65.0	2219.97	64.2
Average	%	1.09 %	178.1	0.19 %	0.43 %	49.9	0.45 %	118.6	0.35 %	125.4
G-01	**14104.64**	14388.58	651							
G-02	**19111.60**	19505.00	1178							
G-03	**24308.19**	24978.17	2061							
G-04	**34022.19**	34957.98	3027							
G-05	14223.63	14683.03	589							
G-06	**21357.16**	22260.19	1021							
G-07	**23251.53**	23963.36	1628							
G-08	**29597.19**	30496.18	2419							
G-09	**1318.03**	1341.17	832	1323.57	1324.87	274.7	1328.14	611.0	1325.62	819.1
G-10	**1581.68**	1612.09	1294	1592.93	1598.14	471.2	1590.83	938.8	1590.82	1762.3
G-11	**2155.67**	2198.45	2004	2166.66	2174.45	720.2	2172.28	1492.7	2173.80	3284.3
G-12	**2472.49**	2521.79	2900	2490.01	2499.80	1071.4	2492.75	2309.7	2495.02	8587.6
G-13	**2256.60**	2286.91	802	2271.29	2274.13	157.6	2278.99	360.8	2274.12	504.5
G-14	**2678.19**	2750.75	1251	2693.35	2702.50	248.4	2705.00	610.4	2703.31	976.9
G-15	**3143.29**	3216.99	1862	3157.31	3162.85	377.1	3158.92	924.8	3161.26	1952.0
G-16	**3606.30**	3693.62	2778	3637.52	3645.80	556.4	3639.11	1313.7	3638.39	4675.1
G-17	1666.31	1701.58	806							
G-18	**2727.80**	2765.92	1303							
G-19	**3491.54**	3576.92	1903							
G-20	**4295.00**	4378.13	2800							
Average	%	2.38 %	1655.5	*0.61 %*	*0.87 %*	*484.6*	*0.79 %*	*1070.2*	*0.76 %*	*2820.2*
TotalAverage	%	1.85 %	1047.1	*0.34 %*	*0.59 %*	*208.0*	*0.58 %*	*464.7*	*0.50 %*	*1105.3*

Table B.9: Comparison of heuristics in the literature on homogenous instances with original outsource costs: Part 1.
Note that the results from the Tabu Searches of Côté and Potvin (2009) (TS25) and Potvin and Naud (2011) (TS and TS+) are omitted for G-(H-)-01 up to G-(H-)-07 and from G-(H-)-17 to G-(H-)-20. This is because the heuristics of Côté and Potvin (2009) and Potvin and Naud (2011) accidently truncate the coordinates to integers when loading the instances. The truncation implies that those instances are different and hence they cannot be compared.

		GA Kratica et al. (2012)		AVNS Stenger et al. (2013)		AVNS-RN Stenger et al. (2013)		
Instance	BKS	best	CPU(s)	best	CPU(s)	best	avg	CPU(s)
CE-01	1119.47	1158.98	49.7	1123.95	92.5	1119.47	1124.06	81.2
CE-02	1814.52	1893.66	91.2	1814.52	48.6	1814.52	1816.88	63.4
CE-03	1919.05	1987.75	111.4	1920.86	212.1	1919.05	1931.52	258.1
CE-04	2505.39	2668.87	204.1	2512.05	279.7	2509.20	2526.00	179.6
CE-05	**3081.42**	3279.64	342.1	3099.77	228.6	3111.61	3121.21	132.9
CE-06	1207.47	1233.20	47.3	1207.81	75.9	1207.47	1208.80	79.8
CE-07	2004.53	2086.17	92.0	2013.93	50.9	2004.53	2008.74	61.0
CE-08	2052.05	2130.82	113.8	2052.05	253.1	2052.05	2062.31	251.1
CE-09	**2418.64**	2558.70	224.7	2432.51	259.0	2431.22	2439.94	190.8
CE-10	**3373.42**	3598.36	358.7	3391.35	201.0	3389.09	3407.02	162.2
CE-11	2330.94	2383.34	137.4	2332.21	316.0	2330.94	2332.34	370.5
CE-12	1952.86	2042.84	111.0	1953.55	92.9	1953.64	1953.64	107.5
CE-13	2858.83	2929.02	163.7	2858.94	278.5	2858.94	2860.94	351.4
CE-14	2213.02	2338.22	103.8	2215.38	93.2	2215.45	2215.45	148.8
Average	%	4.42 %	153.6	0.23 %	177.3	0.16 %	0.45 %	174.2
G-01	**14104.64**	14910.52	629.3	14157.08	652.6	14155.86	14178.46	979.3
G-02	**19111.60**	20258.91	4568.4	19204.36	1558.4	19187.73	19283.51	1809.6
G-03	**24308.19**	25941.17	13529.3	24602.61	2356.1	24535.87	24706.75	2267.4
G-04	**34022.19**	36083.77	22360.7	34415.82	2500.9	34535.60	34682.98	2241.1
G-05	14223.63	14875.44	1803.1	14272.32	1301.1	14276.20	14318.88	2263.5
G-06	**21357.16**	22440.03	4826.8	21440.79	1783.5	21396.38	21529.04	2049.0
G-07	**23251.53**	24621.42	11098.2	23375.60	2262.8	23434.04	23607.02	2107.7
G-08	**29597.19**	31326.38	12532.0	29797.62	2339.7	29819.94	29920.96	1907.3
G-09	**1318.03**	1368.47	3236.9	1335.45	602.0	1332.04	1341.42	704.1
G-10	**1581.68**	1646.20	7682.2	1604.50	978.4	1604.41	1615.02	864.9
G-11	**2155.67**	2235.24	17381.0	2189.02	1534.3	2188.65	2205.53	1311.0
G-12	**2472.49**	2578.12	32100.7	2520.29	2043.9	2528.72	2542.11	1975.8
G-13	**2256.60**	2347.49	1113.6	2291.83	116.5	2283.74	2294.41	170.8
G-14	**2678.19**	2796.74	2454.8	2708.22	183.5	2717.77	2728.54	215.6
G-15	**3143.29**	3283.07	5083.7	3194.82	357.3	3177.52	3198.96	221.0
G-16	**3606.30**	3804.04	11131.3	3671.34	561.2	3674.68	3689.94	370.0
G-17	1666.31	1898.36	372.9	1682.49	110.3	1672.02	1694.55	182.3
G-18	**2727.80**	3079.03	851.8	2741.80	156.4	2751.63	2761.05	211.5
G-19	**3491.54**	3940.71	1110.9	3507.94	194.1	3516.85	3530.97	234.3
G-20	**4295.00**	4823.76	1606.5	4332.44	290.3	4347.85	4358.94	241.3
Average	%	6.61 %	7773.7	1.02 %	1094.2	1.02 %	1.56 %	1116.4
TotalAverage	%	5.71 %	4636.0	0.69 %	716.6	0.67 %	1.11 %	728.4

Table B.10: Comparison of heuristics in the literature on homogenous instances with original outsource costs: Part 2.

		MS-LS Vidal et al. (2015)			MS-ILS Vidal et al. (2015)			UHGS Vidal et al. (2015)		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-01	1119.47	1121.32	1128.28	5.2	1119.47	1119.66	34.1	1119.47	1119.66	38.5
CE-02	1814.52	1840.71	1883.44	3.9	1814.52	1817.34	68.8	1814.52	1815.63	59.6
CE-03	1919.05	1943.64	1958.80	17.0	1922.18	1929.60	258.1	1919.05	1922.88	476.8
CE-04	2505.39	2548.29	2568.49	26.3	2505.39	2516.12	576.7	2505.39	2509.82	934.7
CE-05	**3081.42**	3181.86	3201.29	29.3	3090.53	3102.95	620.7	3081.59	3095.58	1289.3
CE-06	1207.47	1207.47	1216.57	5.8	1207.47	1207.56	36.5	1207.47	1207.47	38.5
CE-07	2004.53	2046.62	2079.67	4.1	2006.52	2022.93	71.2	2006.52	2012.33	73.2
CE-08	2052.05	2088.10	2100.59	17.3	2054.64	2062.21	263.4	2052.05	2057.57	500.3
CE-09	**2418.64**	2478.01	2505.24	25.9	2428.03	2433.28	482.1	2424.32	2428.19	1241.0
CE-10	**3373.42**	3462.56	3491.59	30.1	3382.23	3393.78	714.9	3381.67	3387.12	1229.9
CE-11	2330.94	2343.03	2408.13	37.5	2330.94	2336.06	891.7	2330.94	2331.13	1202.9
CE-12	1952.86	1970.05	1982.06	8.7	1952.86	1953.13	113.6	1952.86	1953.13	150.8
CE-13	2858.83	2909.83	3025.26	40.1	2858.83	2859.01	881.6	2858.83	2859.07	1184.9
CE-14	2213.02	2215.38	2226.44	9.7	2213.02	2213.02	144.8	2213.02	2213.02	189.8
Average	%	1.44 %	2.73 %	18.6	0.10 %	0.34 %	368.4	0.04 %	0.17 %	615.0
G-01	**14104.64**	14272.27	14329.96	102.5	14151.74	14165.45	1811.6	14131.18	14151.51	2405.9
G-02	**19111.60**	19417.12	19524.50	209.6	19142.75	19191.56	1828.1	19166.58	19190.77	2409.9
G-03	**24308.19**	24916.33	25038.41	415.2	24493.16	24609.36	1828.8	24409.02	24588.29	2418.1
G-04	**34022.19**	34883.27	35182.78	639.2	34708.93	34907.49	1864.6	34362.80	34517.47	2421.1
G-05	14223.63	14492.24	14735.12	199.1	14255.09	14373.87	1817.5	14223.63	14296.07	2408.0
G-06	**21357.16**	21741.15	22024.07	285.3	21382.16	21546.18	1834.2	21396.60	21488.29	2411.4
G-07	**23251.53**	23751.10	23980.00	381.3	23407.50	23547.12	1830.5	23373.38	23463.05	2414.9
G-08	**29597.19**	30271.82	30459.11	447.7	29953.21	30064.28	1856.4	29823.18	29918.06	2415.7
G-09	**1318.03**	1370.26	1397.08	60.7	1332.09	1339.06	1305.5	1328.65	1332.63	2323.4
G-10	**1581.68**	1664.96	1682.31	95.9	1595.45	1617.58	1709.2	1597.61	1603.82	2342.3
G-11	**2155.67**	2248.04	2281.79	163.3	2196.75	2228.23	1811.6	2182.01	2192.68	2405.2
G-12	**2472.49**	2624.19	2652.57	236.9	2540.92	2553.40	1818.0	2522.64	2529.84	2407.2
G-13	**2256.60**	2319.74	2337.43	29.3	2274.19	2277.57	513.1	2258.02	2261.50	1412.7
G-14	**2678.19**	2764.11	2791.23	43.5	2701.78	2708.56	917.7	2683.73	2687.50	1935.7
G-15	**3143.29**	3272.34	3296.86	61.8	3170.50	3177.53	1480.4	3145.11	3152.00	2301.4
G-16	**3606.30**	3794.17	3811.80	89.7	3641.69	3672.62	1755.0	3620.71	3632.04	2450.1
G-17	1666.31	1708.26	1717.55	19.4	1669.59	1677.37	379.0	1666.31	1671.72	1805.3
G-18	**2727.80**	2793.63	2801.70	27.7	2734.81	2741.10	582.7	2730.55	2733.12	2035.0
G-19	**3491.54**	3570.94	3585.64	36.6	3508.53	3515.47	712.9	3497.20	3504.26	1989.5
G-20	**4295.00**	4405.90	4433.41	45.9	4316.28	4333.59	1079.2	4312.45	4319.37	2523.0
Average	%	3.03 %	3.93 %	179.5	0.85 %	1.39 %	1436.8	0.49 %	0.80 %	2261.8
TotalAverage	%	2.38 %	3.43 %	113.3	0.54 %	0.95 %	996.9	0.31 %	0.54 %	1583.7

Table B.11: Comparison of heuristics in the literature on homogenous instances with original outsource costs: Part 3.
In the paper of Vidal et al. (2015), the running times are not given. These times were provided by him by email.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-01	1119.47	1119.47	1120.15	10.1	1119.47	1119.47	16.2	1119.47	1119.47	47.4
CE-02	1814.52	1814.52	1821.07	24.0	1814.52	1817.89	40.5	1814.52	1817.06	95.8
CE-03	1919.05	1921.20	1930.42	50.9	1921.10	1928.96	94.0	1919.05	1925.08	236.7
CE-04	2505.39	2520.88	2529.49	122.4	2506.40	2521.85	265.1	2509.81	2518.14	642.7
CE-05	**3081.42**	3096.61	3110.45	234.5	3086.88	3102.68	496.4	3090.49	3101.40	1437.2
CE-06	1207.47	1207.47	1208.43	11.2	1207.47	1207.56	15.0	1207.47	1207.47	44.2
CE-07	2004.53	2004.88	2011.64	26.1	2004.53	2009.03	33.0	2004.53	2009.58	98.6
CE-08	2052.05	2065.16	2072.62	51.0	2058.22	2068.52	111.4	2052.05	2059.56	239.6
CE-09	**2418.64**	2438.35	2447.73	134.0	2425.41	2431.26	309.2	2420.71	2426.35	953.9
CE-10	**3373.42**	3399.95	3413.53	219.5	3380.43	3392.14	554.5	3373.84	3388.22	1704.1
CE-11	2330.94	2330.94	2392.91	68.7	2330.94	2338.09	132.1	2330.94	2330.94	399.0
CE-12	1952.86	1953.55	1966.32	47.5	1952.86	1953.13	85.1	1952.86	1952.86	183.6
CE-13	2858.83	2858.94	2892.05	75.8	2858.83	2859.19	130.9	2858.83	2858.92	437.8
CE-14	2213.02	2214.11	2214.32	42.2	2213.02	2214.24	55.2	2213.02	2213.78	185.2
Average	%	0.25 %	0.81 %	79.8	0.08 %	0.32 %	167.0	0.04 %	0.22 %	479.0
G-01	**14104.64**	14215.96	14255.60	327.2	14159.16	14178.79	754.8	14129.48	14163.43	2433.9
G-02	**19111.60**	19427.81	19570.07	701.9	19145.09	19212.59	1573.0	19140.69	19254.23	6630.0
G-03	**24308.19**	24815.06	25175.22	1282.0	24435.85	24584.51	3224.4	24406.67	24566.00	11062.8
G-04	**34022.19**	34650.73	35012.77	1853.4	34285.64	34485.85	4939.1	34231.56	34425.00	15875.4
G-05	14223.63	14398.20	14627.22	311.4	14223.63	14305.10	414.3	14229.50	14261.06	1591.3
G-06	**21357.16**	21624.94	21979.24	656.2	21509.52	21564.90	825.5	21357.16	21440.38	4337.2
G-07	**23251.53**	23807.31	24096.44	1039.2	23452.22	23570.53	2156.5	23263.22	23440.51	7585.3
G-08	**29597.19**	29960.57	30181.65	1694.4	29717.00	29874.71	3917.1	29657.38	29864.19	12316.5
G-09	**1318.03**	1323.25	1328.30	458.9	1321.20	1324.08	1211.1	1320.29	1325.79	3852.0
G-10	**1581.68**	1586.47	1595.64	684.9	1583.78	1590.01	2096.5	1588.05	1592.14	6922.9
G-11	**2155.67**	2167.57	2180.98	1203.8	2164.63	2173.27	3850.1	2163.50	2172.45	12303.3
G-12	**2472.49**	2493.57	2511.72	2074.2	2480.21	2493.22	7812.0	2483.06	2493.94	19555.0
G-13	**2256.60**	2273.85	2284.81	218.3	2266.43	2272.29	523.2	2261.66	2264.92	2260.2
G-14	**2678.19**	2694.86	2710.24	396.6	2694.34	2699.00	1163.7	2684.66	2689.99	4838.5
G-15	**3143.29**	3168.79	3174.45	1014.1	3151.41	3161.40	2344.9	3150.67	3156.84	8198.0
G-16	**3606.30**	3641.54	3683.87	1086.1	3633.69	3645.61	3558.9	3624.56	3633.76	12741.7
G-17	1666.31	1676.52	1683.68	365.5	1666.96	1674.11	782.5	1666.31	1672.81	1878.0
G-18	**2727.80**	2737.70	2749.69	604.5	2735.65	2739.57	1144.7	2731.28	2734.86	3916.6
G-19	**3491.54**	3530.26	3538.79	961.2	3502.05	3509.68	1943.4	3494.28	3499.55	5225.4
G-20	**4295.00**	4347.73	4375.36	1267.8	4320.54	4331.77	2750.6	4307.63	4314.48	8081.6
Average	%	1.05 %	1.82 %	910.1	0.41 %	0.77 %	2349.3	0.24 %	0.60 %	7580.3
TotalAverage	%	0.72 %	1.40 %	568.2	0.27 %	0.59 %	1450.7	0.16 %	0.44 %	4656.2

Table B.12: Comparison of heuristics in the literature on homogenous instances with original outsource costs: Part 4.

		RIP Bolduc et al. (2008)		TS Potvin and Naud (2011)		TS+ Potvin and Naud (2011)		GA Kratica et al. (2012)	
Instance	BKS	single	CPU(s)	best	CPU(s)	best	CPU(s)	best	CPU(s)
CE-H-01	1191.70	1192.72	26.0	1191.70	25.7	1191.70	26.0	1203.27	49.2
CE-H-02	**1789.41**	1798.26	72.0	1795.51	33.7	1791.21	34.8	1860.84	94.4
CE-H-03	**1916.81**	1934.85	105.0	1926.33	79.0	1917.96	80.8	1988.73	109.3
CE-H-04	**2459.56**	2493.93	251.0	2481.64	195.6	2481.68	198.5	2622.24	206.1
CE-H-05	**3116.71**	3195.66	490.0	3143.92	295.9	3143.01	342.4	3314.16	340.4
CE-H-06	1204.48	1210.23	25.0	1206.82	25.4	1206.82	25.1	1210.75	51.5
CE-H-07	2025.98	2042.79	74.0	2035.90	32.5	2031.85	32.0	2108.23	90.4
CE-H-08	**1983.27**	2015.72	112.0	1991.23	81.4	1986.51	84.5	2057.75	119.2
CE-H-09	**2419.69**	2445.88	267.0	2445.49	188.9	2447.58	193.0	2601.96	233.7
CE-H-10	**3246.91**	3304.69	482.0	3271.70	309.5	3272.37	342.4	3415.40	382.5
CE-H-11	**2301.78**	2308.76	188.0	2325.74	127.0	2336.51	133.9	2381.52	139.5
CE-H-12	**1908.05**	1908.74	130.0	1912.47	60.7	1915.05	60.4	1954.80	109.7
CE-H-13	**2832.88**	2842.18	195.0	2872.14	125.0	2868.13	136.5	2883.67	143.9
CE-H-14	1907.74	1920.36	114.0	1925.46	65.8	1907.75	67.2	1988.79	111.6
Average		0.90 %	180.8	0.65 %	117.6	0.54 %	125.5	3.91 %	155.8
G-H-01	**14084.33**	14408.31	647.0					14812.40	661.9
G-H-02	**18348.97**	18663.15	1254.0					19395.20	4757.0
G-H-03	**24897.32**	25561.55	2053.0					26523.43	1404.3
G-H-04	**34033.16**	35495.66	2904.0					36261.53	23744.7
G-H-05	**15423.65**	16138.50	512.0					16254.20	889.6
G-H-06	**19640.47**	20329.04	1005.0					20717.86	5350.7
G-H-07	**23244.67**	24184.83	1608.0					24727.21	10955.3
G-H-08	**27053.10**	27710.66	2584.0	27521.28	3408.2	27334.84	18625.2	28605.47	23568.1
G-H-09	**1318.74**	1346.03	814.0	1331.11	592.7	1329.27	1829.3	1386.03	3259.7
G-H-10	**1543.97**	1575.82	1332.0	1554.96	1087.1	1555.59	1564.3	1622.14	8750.4
G-H-11	**2172.77**	2218.91	2140.0	2191.23	1445.5	2195.83	3207.9	2266.04	14759.3
G-H-12	**2462.95**	2510.07	2970.0	2535.00	2108.3	2482.92	4224.0	2580.32	32527.2
G-H-13	**2215.19**	2253.45	733.0	2231.88	405.8	2237.38	1801.3	2330.81	1256.3
G-H-14	**2648.26**	2711.81	1246.0	2685.51	630.3	2684.70	1042.9	2809.86	2687.7
G-H-15	**3099.16**	3156.93	1895.0	3123.60	976.6	3127.33	2111.2	3285.70	5963.1
G-H-16	**3586.62**	3649.09	2785.0	3853.21	1571.2	3621.85	6217.4	3780.43	10786.4
G-H-17	**1688.66**	1705.48	762.0					1932.18	389.1
G-H-18	**2730.52**	2759.99	1299.0					3062.08	729.9
G-H-19	**3442.57**	3517.48	1892.0					3892.96	1004.0
G-H-20	**4306.64**	4413.82	2733.0					4865.32	1450.4
Average		2.41 %	1658.4	*1.95 %*	*1358.4*	*0.97 %*	*4513.7*	7.08 %	7744.7
TotalAverage		1.79 %	1050.0	*1.16 %*	*603.1*	*0.71 %*	*1842.7*	5.77 %	4619.9

Table B.13: Comparison of heuristics in the literature on heterogenous instances with original outsource costs: Part 1.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-H-01	1191.70	1191.70	1193.00	10.1	1191.70	1194.22	15.3	1191.70	1193.09	53.0
CE-H-02	**1789.41**	1795.91	1804.13	25.2	1796.13	1802.79	39.4	1789.41	1799.19	129.1
CE-H-03	**1916.81**	1922.82	1936.32	52.4	1916.81	1927.28	82.9	1917.46	1924.47	236.2
CE-H-04	**2459.56**	2484.79	2499.79	128.6	2477.69	2491.28	180.3	2468.63	2480.63	916.8
CE-H-05	**3116.71**	3152.15	3167.93	257.3	3135.83	3150.62	529.8	3123.07	3136.80	1682.0
CE-H-06	1204.48	1205.77	1209.52	11.9	1206.82	1211.36	16.0	1204.48	1207.75	59.9
CE-H-07	2025.98	2034.98	2041.41	24.6	2033.42	2039.31	44.2	2025.98	2032.50	119.8
CE-H-08	**1983.27**	1987.54	2004.60	53.1	1989.21	1995.29	92.2	1983.96	1987.58	326.7
CE-H-09	**2419.69**	2461.18	2471.66	129.5	2444.36	2453.97	264.5	2425.06	2446.58	1000.2
CE-H-10	**3246.91**	3271.86	3288.80	245.5	3267.88	3281.33	429.9	3254.66	3262.88	1937.1
CE-H-11	**2301.78**	2365.32	2418.32	88.7	2312.92	2371.09	144.2	2305.73	2313.60	527.6
CE-H-12	**1908.05**	1925.45	1937.72	50.9	1908.93	1934.57	65.3	1908.05	1919.17	192.8
CE-H-13	**2832.88**	2844.41	2925.09	77.2	2836.09	2863.02	104.0	2833.77	2847.97	404.7
CE-H-14	1907.74	1928.71	1948.96	51.4	1916.45	1929.76	72.6	1913.55	1925.31	198.9
Average		0.80 %	1.64 %	86.2	0.38 %	1.06 %	148.6	0.11 %	0.54 %	556.1
G-H-01	**14084.33**	14220.30	14315.62	332.8	14175.86	14197.77	643.2	14084.33	14123.48	3608.8
G-H-02	**18348.97**	18561.79	18737.67	699.9	18482.03	18533.04	2242.3	18403.41	18523.85	8184.2
G-H-03	**24897.32**	25315.53	25740.72	1119.7	25049.48	25209.63	2775.0	24949.88	25127.31	11445.1
G-H-04	**34033.16**	34654.94	35062.09	1679.5	34362.17	34476.52	4289.3	34147.04	34405.37	14821.2
G-H-05	**15423.65**	15647.09	15879.83	249.4	15423.65	15565.58	358.9	15423.65	15491.65	1619.8
G-H-06	**19640.47**	20030.05	20275.03	549.4	19742.57	19905.66	1161.2	19695.40	19868.59	4414.2
G-H-07	**23244.67**	23893.59	24162.40	886.0	23532.62	23638.61	2420.2	23394.23	23544.57	9853.6
G-H-08	**27053.10**	27446.57	27738.53	1452.9	27155.11	27305.97	4593.2	27111.30	27343.98	17610.4
G-H-09	**1318.74**	1346.36	1356.71	403.6	1330.30	1344.25	1060.9	1325.09	1330.04	4634.4
G-H-10	**1543.97**	1582.04	1591.64	724.0	1568.51	1575.60	1887.8	1551.92	1560.45	7680.1
G-H-11	**2172.77**	2216.85	2227.39	1110.8	2199.48	2216.19	3570.7	2188.14	2197.42	10844.3
G-H-12	**2462.95**	2539.58	2547.00	1773.3	2496.50	2519.29	8742.8	2491.08	2510.01	22103.2
G-H-13	**2215.19**	2260.02	2266.93	243.0	2229.22	2243.92	921.0	2216.80	2227.53	2819.4
G-H-14	**2648.26**	2690.26	2726.82	343.0	2660.89	2675.40	1646.0	2653.58	2664.39	5595.0
G-H-15	**3099.16**	3147.99	3168.74	853.5	3134.03	3148.68	2402.7	3111.08	3124.36	8458.0
G-H-16	**3586.62**	3672.97	3682.11	1334.9	3621.89	3649.42	4877.1	3608.27	3624.19	14078.1
G-H-17	**1688.66**	1705.92	1716.15	305.5	1696.72	1700.91	1068.0	1689.30	1695.24	2661.4
G-H-18	**2730.52**	2757.40	2775.02	512.1	2738.17	2752.74	1741.4	2736.29	2740.59	3760.1
G-H-19	**3442.57**	3505.20	3534.22	841.5	3461.47	3486.10	2570.1	3451.70	3462.83	7257.6
G-H-20	**4306.64**	4399.10	4435.14	1167.4	4345.29	4364.39	3737.8	4321.06	4341.49	10005.3
Average		1.83 %	2.69 %	829.1	0.78 %	1.37 %	2635.5	0.35 %	0.87 %	8572.7
TotalAverage		1.41 %	2.26 %	523.2	0.62 %	1.24 %	1611.5	0.25 %	0.73 %	5271.7

Table B.14: Comparison of heuristics in the literature on heterogenous instances with original outsource costs: Part 2.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-01	940.66	940.66	940.66	8.2	940.66	940.66	18.4	940.66	940.66	53.4
CE-02	1605.51	1605.51	1608.00	15.1	1605.51	1607.48	26.1	1605.51	1607.05	122.3
CE-03	1706.18	1706.18	1723.93	28.4	1706.18	1706.78	58.8	1706.18	1706.66	243.1
CE-04	2237.96	2247.35	2259.89	97.6	2245.37	2250.13	189.3	2237.96	2243.47	515.5
CE-05	2770.34	2795.24	2818.86	149.0	2777.37	2791.48	328.7	2770.34	2778.85	1481.1
CE-06	985.66	985.66	985.66	7.8	985.66	985.66	18.6	985.66	985.66	63.0
CE-07	1721.33	1721.33	1722.70	14.3	1721.33	1721.33	26.5	1721.33	1721.33	129.7
CE-08	1804.49	1804.68	1808.73	25.4	1804.49	1804.49	61.7	1804.49	1804.49	237.1
CE-09	2250.46	2259.03	2272.57	91.4	2254.69	2261.07	166.3	2250.68	2255.55	807.6
CE-10	3024.15	3037.10	3051.76	152.8	3025.90	3034.87	426.9	3024.36	3031.39	1361.9
CE-11	2172.90	2172.90	2187.15	45.6	2172.90	2172.90	64.9	2172.90	2172.90	439.7
CE-12	1800.85	1800.85	1817.46	36.9	1800.85	1802.52	57.2	1800.85	1800.85	164.4
CE-13	2626.79	2626.79	2641.49	49.5	2626.79	2626.79	58.5	2626.79	2626.79	385.2
CE-14	1922.85	1932.46	1933.59	34.7	1922.85	1925.38	47.2	1922.85	1927.92	203.9
Average		0.19 %	0.63 %	54.0	0.06 %	0.18 %	110.6	0.00 %	0.10 %	443.4
G-01	12436.61	12515.67	12575.64	380.2	12443.83	12473.49	584.6	12436.61	12470.22	1805.9
G-02	17776.73	18085.95	18177.09	723.1	17785.73	17888.16	1449.8	17798.71	17851.93	6159.2
G-03	23059.15	23542.31	23791.36	1380.6	23059.15	23161.63	3802.2	23093.44	23187.79	10620.9
G-04	30502.95	31401.51	31643.91	2127.9	30502.95	30743.75	6024.1	30582.02	30701.44	15880.1
G-05	13504.63	13620.64	13808.72	274.0	13549.86	13565.68	442.0	13510.42	13539.15	1203.6
G-06	18804.73	19131.60	19316.61	589.7	18810.01	18943.16	1356.5	18804.73	18904.76	3196.8
G-07	21676.11	22198.97	22338.11	1047.8	21676.11	21835.66	3004.9	21701.00	21759.72	8149.5
G-08	26262.70	26795.63	26956.80	1639.6	26262.70	26437.72	3838.7	26270.65	26369.04	11898.5
G-09	1208.17	1210.49	1215.13	357.1	1210.15	1213.28	1017.3	1208.17	1210.48	3028.9
G-10	1459.31	1467.58	1472.40	619.0	1461.32	1467.08	2686.0	1459.31	1463.96	5932.9
G-11	1989.39	2003.58	2008.89	1011.5	1990.65	2001.80	3357.5	1992.44	1999.34	10556.3
G-12	2308.00	2331.91	2337.43	1594.8	2313.66	2326.35	4292.4	2308.00	2321.32	16791.0
G-13	1864.50	1871.21	1876.80	295.2	1867.66	1873.54	660.7	1864.50	1869.14	2382.9
G-14	2273.69	2278.74	2289.17	538.8	2275.42	2282.32	1190.6	2273.69	2279.73	3939.8
G-15	2727.13	2742.70	2747.59	803.1	2728.51	2736.12	2371.4	2727.13	2733.24	6804.9
G-16	3194.65	3214.05	3220.45	1425.5	3201.93	3209.86	3811.3	3194.65	3203.14	11057.4
G-17	1506.42	1514.64	1520.32	244.5	1508.83	1510.51	447.1	1506.42	1508.23	1794.6
G-18	2420.46	2430.18	2436.72	520.7	2424.70	2426.59	840.5	2420.46	2422.93	3122.4
G-19	3099.49	3118.98	3131.01	756.3	3106.97	3112.98	1618.6	3099.49	3105.88	5261.1
G-20	3886.86	3915.36	3923.88	986.0	3888.46	3898.38	2551.2	3886.86	3892.43	8428.8
Average		1.05 %	1.56 %	865.8	0.11 %	0.50 %	2267.4	0.04 %	0.33 %	6900.8
TotalAverage		0.70 %	1.19 %	532.9	0.09 %	0.37 %	1382.0	0.03 %	0.24 %	4253.3

Table B.15: Comparison of our heuristics on the homogeneous instances with half-original outsource costs.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-H-01	977.38	977.38	983.18	8.4	977.38	977.38	16.9	977.38	977.38	52.3
CE-H-02	1583.65	1585.32	1592.20	18.9	1585.32	1585.47	33.4	1585.32	1585.33	147.1
CE-H-03	1688.51	1708.63	1724.24	30.0	1688.51	1692.05	48.2	1688.51	1691.50	234.8
CE-H-04	2205.69	2214.48	2250.90	90.0	2207.15	2213.60	199.3	2205.69	2213.08	729.9
CE-H-05	2791.55	2822.16	2854.05	161.0	2804.41	2815.80	310.0	2791.55	2801.69	1662.1
CE-H-06	977.38	977.38	986.10	7.6	977.38	977.38	12.6	977.38	977.38	58.3
CE-H-07	1704.52	1713.37	1732.40	15.8	1704.52	1705.44	27.2	1704.52	1704.52	129.0
CE-H-08	1789.60	1796.14	1814.53	31.9	1790.59	1795.16	55.5	1789.60	1792.03	312.4
CE-H-09	2229.19	2260.34	2282.47	94.3	2236.01	2244.82	166.6	2232.48	2238.13	741.2
CE-H-10	2958.74	2992.47	3020.20	164.1	2968.81	2977.85	359.3	2958.74	2966.11	1644.8
CE-H-11	2144.23	2163.30	2232.65	52.6	2144.23	2164.79	81.4	2146.69	2167.80	312.0
CE-H-12	1759.78	1784.74	1804.19	39.1	1759.78	1768.62	53.5	1759.78	1765.97	179.7
CE-H-13	2611.25	2625.99	2666.18	52.0	2611.25	2617.35	71.3	2611.25	2616.82	310.1
CE-H-14	1742.97	1749.66	1766.54	34.2	1742.97	1744.69	43.6	1742.97	1743.13	201.8
Average		0.68 %	1.86 %	57.1	0.10 %	0.36 %	105.6	0.03 %	0.25 %	479.7
G-H-01	12388.53	12468.29	12562.88	327.7	12388.53	12458.77	812.6	12388.67	12450.85	2214.1
G-H-02	17647.94	17911.18	18015.02	683.1	17680.53	17755.04	1846.4	17647.94	17712.64	6188.3
G-H-03	23185.22	23705.36	23837.41	1153.8	23196.85	23365.56	3637.1	23217.38	23303.71	12854.1
G-H-04	30562.87	31515.65	31754.33	1719.2	30562.87	30745.98	5750.9	30595.08	30727.85	15815.3
G-H-05	13628.09	13989.77	14088.29	209.6	13737.34	13757.65	387.7	13628.09	13708.55	1111.3
G-H-06	18657.87	19139.17	19309.62	492.1	18698.61	18820.31	1153.7	18657.87	18732.83	4321.8
G-H-07	21667.02	22307.68	22410.60	878.2	21677.33	21818.77	2937.1	21667.02	21755.58	8852.1
G-H-08	25605.72	26220.00	26373.02	1425.8	25605.72	25825.78	4948.0	25701.52	25846.74	15225.8
G-H-09	1189.89	1196.27	1219.74	317.2	1191.10	1199.68	1206.1	1189.89	1196.42	3085.1
G-H-10	1419.46	1430.80	1444.11	588.2	1419.67	1426.49	2272.4	1419.46	1425.22	5763.5
G-H-11	1994.00	2003.06	2028.53	984.7	1998.34	2016.69	3670.1	1994.00	2004.07	11671.8
G-H-12	2294.19	2319.86	2349.60	1402.3	2296.31	2316.75	7551.0	2294.19	2304.20	16820.4
G-H-13	1822.11	1836.60	1850.56	261.0	1824.70	1831.04	770.5	1822.11	1827.18	1929.4
G-H-14	2216.72	2234.72	2253.12	468.5	2217.67	2225.52	1826.2	2216.72	2224.22	3650.2
G-H-15	2665.05	2690.17	2710.19	760.4	2665.05	2675.88	3163.2	2668.01	2675.72	6414.7
G-H-16	3133.58	3158.71	3183.54	1237.2	3138.99	3144.90	4869.7	3133.58	3144.02	10913.1
G-H-17	1484.19	1496.98	1509.74	234.7	1490.03	1494.31	508.3	1484.19	1488.67	2381.9
G-H-18	2389.68	2416.87	2443.22	419.7	2399.25	2403.82	1343.3	2389.68	2393.91	4120.3
G-H-19	3039.23	3104.18	3118.94	670.9	3052.30	3063.66	1937.3	3039.23	3049.80	6645.9
G-H-20	3832.45	3901.84	3930.85	981.8	3850.45	3865.79	3070.5	3832.45	3847.81	8697.7
Average		1.51 %	2.38 %	760.8	0.19 %	0.70 %	2683.1	0.04 %	0.44 %	7433.8
TotalAverage		1.19 %	2.20 %	472.5	0.15 %	0.57 %	1624.4	0.03 %	0.37 %	4582.9

Table B.16: Comparison of our heuristics on the heterogenous instances with half-original outsource costs.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-01	1082.37	1082.37	1084.47	6.7	1082.37	1083.71	11.1	1082.37	1082.37	68.7
CE-02	1764.02	1769.18	1772.25	14.3	1764.02	1768.74	24.7	1764.02	1765.48	143.0
CE-03	1779.45	1787.64	1796.86	30.2	1783.00	1785.13	64.7	1782.66	1783.89	242.2
CE-04	2290.51	2305.44	2319.44	80.9	2293.07	2301.22	180.7	2291.79	2294.44	736.7
CE-05	2780.37	2788.17	2799.96	146.5	2784.57	2792.04	235.1	2780.37	2784.69	1232.4
CE-06	1202.26	1202.26	1208.13	7.0	1202.26	1202.62	12.6	1202.26	1202.26	36.7
CE-07	1983.37	1987.44	1997.00	17.4	1983.37	1987.70	37.7	1983.37	1983.52	117.4
CE-08	1948.95	1950.02	1966.43	32.2	1948.95	1952.61	78.0	1948.95	1949.99	185.9
CE-09	2310.51	2330.64	2341.48	67.0	2318.01	2325.42	183.7	2310.51	2318.37	606.9
CE-10	3150.98	3157.35	3171.65	182.5	3158.20	3165.19	407.3	3152.73	3157.29	1105.9
CE-11	2203.91	2204.53	2211.57	45.8	2204.53	2205.18	95.2	2205.16	2205.38	363.0
CE-12	1894.10	1894.10	1902.76	36.7	1894.10	1896.49	52.8	1894.10	1894.10	172.5
CE-13	2857.77	2857.91	2865.42	79.3	2857.77	2861.62	119.8	2857.77	2859.08	333.4
CE-14	2083.85	2091.04	2117.77	21.7	2083.85	2084.82	75.7	2083.85	2084.95	175.8
Average		0.24 %	0.74 %	54.9	0.07 %	0.25 %	112.8	0.02 %	0.10 %	394.3
G-01	12160.83	12167.03	12203.75	274.3	12171.99	12199.00	578.0	12160.83	12172.32	2209.0
G-02	17255.76	17305.88	17340.51	536.4	17277.95	17310.63	1358.9	17272.27	17281.46	5784.7
G-03	22469.01	22538.27	22578.90	975.3	22501.25	22524.09	2197.0	22469.01	22483.14	9927.6
G-04	29646.47	29724.59	29782.63	1496.0	29688.87	29719.40	4954.5	29646.47	29695.83	14295.0
G-05	13660.87	13775.64	13809.26	237.2	13668.69	13678.37	524.7	13660.87	13666.51	2032.7
G-06	18939.58	18988.28	19006.62	425.4	18941.45	18958.08	1179.8	18939.58	18959.35	4827.7
G-07	21258.96	21274.59	21344.67	806.7	21266.90	21309.52	2225.0	21258.96	21279.35	6528.5
G-08	25245.47	25342.60	25379.77	1181.7	25279.78	25314.52	3161.7	25245.47	25305.86	10793.8
G-09	1425.30	1427.05	1440.65	354.3	1427.97	1432.77	945.2	1425.30	1429.17	1980.7
G-10	1680.21	1690.15	1696.69	592.4	1683.72	1687.41	2070.6	1680.21	1684.13	4355.4
G-11	2337.93	2364.69	2378.77	998.9	2350.42	2365.70	3029.8	2337.93	2348.90	8794.1
G-12	2666.87	2696.75	2715.36	1450.7	2679.29	2691.56	5267.5	2666.87	2680.82	11648.1
G-13	2450.61	2456.34	2474.08	320.7	2450.61	2460.97	784.6	2450.88	2456.49	1793.6
G-14	2902.98	2910.44	2933.74	541.2	2907.96	2919.13	1149.7	2902.98	2908.67	3473.8
G-15	3377.13	3408.68	3422.22	856.7	3381.12	3401.22	2007.1	3377.13	3392.05	5030.0
G-16	3864.52	3885.17	3898.93	1213.7	3873.89	3887.29	3777.5	3864.52	3876.37	7499.2
G-17	1504.53	1510.87	1516.40	270.0	1504.59	1508.14	537.3	1504.53	1506.09	1471.8
G-18	2584.71	2601.64	2607.96	455.7	2588.28	2593.54	1440.0	2584.71	2588.30	3604.8
G-19	3187.87	3205.69	3207.08	664.4	3192.01	3199.29	1790.0	3187.87	3191.99	4608.1
G-20	3816.89	3839.71	3847.94	972.3	3826.09	3834.80	2723.2	3816.89	3823.27	6486.3
Average		0.48 %	0.86 %	731.2	0.16 %	0.43 %	2085.1	0.01 %	0.21 %	5857.2
TotalAverage		0.39 %	0.83 %	454.1	0.13 %	0.36 %	1276.0	0.01 %	0.17 %	3617.4

Table B.17: Comparison of our heuristics on the homogenous instances with new outsource costs.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-H-01	1189.28	1190.90	1194.70	7.5	1189.28	1190.69	12.2	1189.28	1189.96	48.7
CE-H-02	1676.73	1676.73	1685.66	16.8	1676.73	1680.68	25.8	1676.73	1677.66	116.2
CE-H-03	1714.44	1732.17	1739.06	38.6	1716.18	1719.61	50.7	1714.44	1719.19	224.6
CE-H-04	2216.66	2220.75	2237.37	90.8	2221.82	2226.70	154.3	2216.66	2222.46	560.8
CE-H-05	2720.67	2738.90	2750.17	159.8	2729.34	2738.32	263.9	2721.72	2730.18	1208.1
CE-H-06	1167.81	1173.22	1174.83	7.0	1168.29	1169.50	13.0	1167.81	1168.06	53.7
CE-H-07	1945.28	1956.60	1963.40	18.0	1945.28	1954.24	29.2	1945.28	1955.11	134.3
CE-H-08	1881.24	1890.61	1900.02	31.1	1883.61	1887.29	62.7	1881.24	1885.15	277.6
CE-H-09	2248.22	2266.16	2281.09	86.8	2261.08	2266.24	150.7	2248.22	2254.21	719.5
CE-H-10	3043.12	3053.87	3070.88	154.1	3045.31	3058.24	307.0	3044.75	3050.71	1440.8
CE-H-11	2087.08	2087.67	2098.38	52.2	2087.43	2088.90	110.9	2087.08	2090.06	443.5
CE-H-12	1810.05	1813.47	1822.05	33.0	1810.56	1815.12	59.0	1810.05	1813.13	279.5
CE-H-13	2785.92	2791.48	2826.10	47.2	2791.48	2802.87	80.6	2791.26	2800.95	464.3
CE-H-14	1764.69	1767.39	1778.25	32.2	1764.69	1766.14	72.7	1764.69	1765.76	220.2
Average		0.38 %	0.91 %	55.4	0.12 %	0.36 %	99.5	0.02 %	0.23 %	442.3
G-H-01	11600.91	11617.17	11687.57	284.3	11608.17	11625.17	731.1	11600.91	11616.10	2280.8
G-H-02	16296.86	16304.96	16444.74	624.9	16296.86	16339.31	1258.0	16297.17	16350.30	5116.4
G-H-03	21192.11	21316.83	21370.25	978.7	21192.11	21226.30	2578.3	21198.02	21219.08	8866.8
G-H-04	27963.45	28043.39	28097.85	1296.1	27963.45	27983.59	3463.7	27963.45	27993.34	11015.6
G-H-05	12978.64	12978.64	13002.80	164.9	12978.64	12978.64	268.1	12978.64	12978.64	1380.4
G-H-06	18744.81	18785.11	18844.10	456.9	18744.81	18775.05	1099.8	18744.81	18764.69	4261.1
G-H-07	20014.52	20058.75	20182.30	741.7	20014.52	20036.97	1886.5	20014.52	20025.81	6477.2
G-H-08	23880.67	23975.58	24050.09	1287.5	23888.10	23953.45	2982.3	23880.67	23938.39	9554.3
G-H-09	1397.97	1417.24	1426.17	309.0	1399.27	1410.64	1139.7	1397.97	1401.84	2576.1
G-H-10	1586.71	1597.32	1601.70	407.5	1588.24	1595.56	1421.8	1586.71	1591.24	3268.0
G-H-11	2324.40	2360.26	2373.82	835.5	2340.72	2350.73	3577.2	2324.40	2340.80	7337.6
G-H-12	2613.29	2653.53	2666.66	882.3	2616.15	2630.18	5505.1	2613.29	2620.57	10575.8
G-H-13	2394.41	2420.39	2426.19	305.1	2395.42	2410.26	685.9	2394.41	2399.65	1986.0
G-H-14	2837.66	2881.24	2892.25	471.7	2847.41	2868.13	1533.7	2837.66	2851.31	3676.1
G-H-15	3285.27	3311.16	3319.57	659.9	3285.27	3297.73	2240.3	3288.63	3295.72	4561.0
G-H-16	3750.00	3758.22	3769.44	933.7	3750.00	3756.17	3406.9	3750.03	3755.35	6594.7
G-H-17	1425.48	1434.22	1440.64	259.1	1427.61	1429.10	612.2	1425.48	1427.18	1631.5
G-H-18	2519.91	2541.13	2547.82	474.2	2523.32	2527.72	967.6	2519.91	2522.79	3641.2
G-H-19	3055.15	3070.46	3087.22	684.5	3060.89	3069.05	1806.5	3055.15	3061.84	5478.8
G-H-20	3669.77	3690.25	3712.09	931.2	3684.28	3692.04	2616.3	3669.77	3680.01	8217.1
Average		0.66 %	1.08 %	649.4	0.12 %	0.42 %	1989.0	0.01 %	0.23 %	5424.8
TotalAverage		0.55 %	1.02 %	406.2	0.12 %	0.40 %	1213.5	0.01 %	0.23 %	3384.8

Table B.18: Comparison of our heuristics on the heterogenous instances with new outsource costs.

		R-TS			LNS-Fast			LNS-Slow		
Instance	BKS	best	avg	CPU(s)	best	avg	CPU(s)	best	avg	CPU(s)
CE-01	524.61	524.61	524.61	9.1	524.61	524.61	9.2	524.61	524.61	22.9
CE-02	835.26	835.26	835.33	23.1	835.26	835.64	33.3	835.26	835.73	72.3
CE-03	826.14	828.56	830.39	42.8	826.14	827.68	70.9	826.14	827.98	194.9
CE-04	1028.42	1036.67	1044.53	101.6	1029.56	1040.04	255.9	1031.07	1036.37	629.2
CE-05	1291.29	1302.89	1312.65	228.6	1296.15	1307.31	470.3	1293.59	1303.11	1431.1
CE-11	1042.12	1042.12	1092.63	64.3	1042.12	1071.29	125.3	1042.12	1069.07	316.7
CE-12	819.56	819.56	819.56	33.1	819.56	819.56	38.1	819.56	819.56	107.2
Average		0.28 %	1.23 %	71.8	0.07 %	0.77 %	143.3	0.06 %	0.65 %	396.3
tai75a	1618.36	1620.47	1650.47	22.3	1618.36	1618.78	27.8	1618.36	1618.36	74.5
tai75b	1344.62	1363.69	1376.85	19.2	1344.64	1344.70	36.2	1344.62	1344.70	88.0
tai75c	1291.01	1335.19	1370.17	23.6	1291.01	1297.42	38.9	1291.01	1291.01	93.0
tai75d	1365.42	1372.13	1392.31	20.3	1365.42	1369.81	35.8	1365.42	1365.42	79.7
tai100a	2041.34	2097.00	2105.62	36.3	2049.58	2073.82	94.3	2041.34	2057.85	268.7
tai100b	1939.90	1948.48	1968.55	43.8	1940.61	1944.33	68.3	1940.61	1940.70	205.0
tai100c	1406.20	1472.52	1523.82	34.7	1415.29	1418.01	73.9	1406.86	1415.51	205.2
tai100d	1581.24	1606.24	1617.13	30.5	1596.97	1598.89	66.1	1592.88	1597.07	228.3
tai150a	3055.23	3096.01	3170.38	107.5	3056.41	3085.33	284.4	3056.85	3059.90	675.4
tai150b	2656.47	2843.75	2888.21	85.5	2732.52	2789.79	226.7	2732.25	2739.43	742.7
tai150c	2341.84	2416.78	2493.59	79.7	2367.22	2405.11	267.9	2365.97	2400.65	616.7
tai150d	2645.39	2755.29	2783.79	71.8	2669.96	2686.74	271.1	2661.69	2668.44	668.0
tai385	24431.44	24914.46	25040.97	875.2	24730.62	24851.31	3687.1	24657.92	24718.72	8218.6
Average		2.51 %	4.19 %	111.6	0.63 %	1.28 %	398.3	0.48 %	0.80 %	935.7
TotalAverage		1.73 %	3.15 %	97.6	0.44 %	1.10 %	309.1	0.34 %	0.75 %	746.9

Table B.19: Comparison of our heuristics on the instances without outsourcing.

Appendix C

Detailed test results of chapter 4

				Initialization				Hall		
	Instance	BFS	Prices	AVNS-Fast	Outsourced	Driven	EstCosts	AVNS-Fast	Outsourced	Driven
Orig	CE-01	1119.47	1964.00	1124.68	71.43 %	95.56 %	1066.31	1119.47	100.00 %	100.00 %
	CE-02	1814.52	3551.00	1814.52	100.00 %	100.00 %	1772.43	1814.53	50.00 %	95.83 %
	CE-03	1919.05	4076.00	1930.74	83.33 %	97.78 %	1868.77	1932.51	83.33 %	97.78 %
	CE-04	2505.39	5869.00	2524.28	76.19 %	96.27 %	2514.25	2525.05	80.00 %	97.01 %
	CE-05	3081.42	7742.00	3115.19	81.48 %	97.18 %	3121.29	3095.25	81.48 %	97.18 %
	CE-06	1207.47	2036.00	1216.09	71.43 %	95.56 %	1153.18	1207.47	100.00 %	100.00 %
	CE-07	2004.53	3737.00	2018.09	80.00 %	98.59 %	1961.15	2004.53	100.00 %	100.00 %
	CE-08	2052.05	4199.00	2052.05	100.00 %	100.00 %	2001.57	2064.45	83.33 %	97.78 %
	CE-09	2418.64	5989.00	2436.65	75.00 %	97.87 %	2433.69	2428.21	90.91 %	99.29 %
	CE-10	3373.42	7985.00	3385.51	81.48 %	97.18 %	3409.09	3394.43	81.48 %	97.18 %
	CE-11	2330.94	8684.00	2377.85	80.00 %	97.22 %	2575.54	2368.38	68.75 %	95.41 %
	CE-12	1952.86	4365.00	1966.73	62.50 %	96.84 %	2169.54	1966.73	62.50 %	96.84 %
	CE-13	2858.83	8829.00	2866.98	62.50 %	94.55 %	3099.23	2891.06	62.50 %	94.55 %
	CE-14	2213.02	4379.00	2215.38	87.50 %	97.67 %	2410.59	2215.38	87.50 %	97.67 %
	Average		129.70 %	0.61 %	79.49 %	97.30 %	4.36 %	0.51 %	80.84 %	97.61 %
Half-Orig	CE-01	940.66	982.00	957.83	17.14 %	34.09 %	921.83	974.56	60.00 %	51.72 %
	CE-02	1605.51	1775.50	1669.33	13.89 %	55.71 %	1615.62	1642.12	70.73 %	73.91 %
	CE-03	1706.18	2038.00	1771.02	34.38 %	76.40 %	1663.92	1773.66	72.73 %	88.16 %
	CE-04	2237.96	2934.50	2262.67	70.37 %	93.89 %	2252.46	2296.98	75.76 %	93.60 %
	CE-05	2770.34	3871.00	2792.47	73.53 %	94.83 %	2811.44	2806.00	79.41 %	95.93 %
	CE-06	985.66	1018.00	1018.00	17.14 %	34.09 %	974.67	1018.00	63.89 %	51.85 %
	CE-07	1721.33	1868.50	1762.57	11.36 %	44.29 %	1735.67	1766.95	76.60 %	71.79 %
	CE-08	1804.49	2099.50	1866.54	35.48 %	77.53 %	1763.54	1833.16	63.16 %	81.58 %
	CE-09	2250.46	2994.50	2310.00	40.74 %	88.49 %	2265.60	2300.02	78.13 %	94.40 %
	CE-10	3023.80	3992.50	3069.13	57.14 %	89.71 %	3067.29	3070.08	84.44 %	95.65 %
	CE-11	2172.90	4342.00	2239.35	41.18 %	81.13 %	2417.02	2176.29	88.57 %	95.51 %
	CE-12	1800.85	2182.50	1856.73	30.43 %	82.80 %	2008.22	1851.36	66.67 %	87.50 %
	CE-13	2626.79	4414.50	2658.57	39.39 %	81.31 %	2869.15	2647.35	86.49 %	94.32 %
	CE-14	1922.85	2189.50	1967.88	58.33 %	88.37 %	2102.25	2003.32	53.49 %	74.03 %
	Average		13.87 %	2.46 %	38.61 %	73.05 %	3.92 %	2.35 %	72.86 %	82.14 %
New	CE-01	1082.37	1169.50	1135.38	25.00 %	73.91 %	1053.09	1134.86	23.53 %	71.74 %
	CE-02	1764.02	2063.50	1808.27	7.14 %	82.43 %	1755.21	1823.21	28.57 %	85.92 %
	CE-03	1779.45	2046.00	1853.73	14.29 %	81.44 %	1771.10	1828.04	25.00 %	84.21 %
	CE-04	2290.51	2663.50	2379.87	19.35 %	82.64 %	2348.15	2340.23	34.48 %	86.43 %
	CE-05	2778.02	3184.75	2873.09	15.38 %	76.96 %	2859.12	2867.88	50.00 %	87.57 %
	CE-06	1202.26	1364.00	1245.38	20.00 %	83.33 %	1162.11	1239.55	20.00 %	83.33 %
	CE-07	1983.37	2474.50	2030.80	16.67 %	93.24 %	1963.58	2022.61	33.33 %	94.52 %
	CE-08	1948.95	2339.50	2013.73	14.29 %	81.44 %	1931.10	1992.20	25.00 %	84.21 %
	CE-09	2310.51	2793.25	2384.40	11.11 %	83.67 %	2356.99	2365.67	32.00 %	88.03 %
	CE-10	3147.93	3819.00	3255.78	20.00 %	83.25 %	3214.71	3202.54	42.86 %	89.13 %
	CE-11	2203.91	2544.75	2261.99	17.07 %	69.91 %	2398.21	2291.01	59.18 %	78.02 %
	CE-12	1894.10	2253.75	1922.22	4.35 %	77.78 %	2074.52	1976.83	44.00 %	84.27 %
	CE-13	2857.77	3682.00	2906.27	18.75 %	88.89 %	3059.34	2917.33	42.86 %	92.98 %
	CE-14	2083.85	2475.50	2121.99	11.54 %	76.29 %	2264.78	2192.32	36.00 %	82.42 %
	Average		28.84 %	3.03 %	15.35 %	81.09 %	3.76 %	3.10 %	35.49 %	85.20 %

Table C.1: Detailed results of the estimation methods: Part 1.

				Baghuis				Fleischmann			
	Instance	BFS	Prices	EstCosts	AVNS-Fast	Outsourced	Driven	EstCosts	AVNS-Fast	Outsourced	Driven
	CE-01	1119.47	1964.00	1155.16	1151.43	55.56 %	91.11 %	972.93	1133.68	71.43 %	95.56 %
	CE-02	1814.52	3551.00	1754.05	1868.50	71.43 %	97.14 %	1612.81	1814.53	50.00 %	95.83 %
	CE-03	1919.05	4076.00	1956.79	1969.38	78.57 %	96.63 %	1683.50	1930.45	83.33 %	97.78 %
	CE-04	2505.39	5869.00	2548.59	2559.71	66.67 %	94.03 %	2282.73	2522.87	80.00 %	97.01 %
	CE-05	3081.42	7742.00	3028.69	3138.83	76.67 %	96.02 %	2845.73	3100.52	81.48 %	97.18 %
	CE-06	1207.47	2036.00	1241.27	1228.46	62.50 %	93.33 %	1059.93	1222.68	71.43 %	95.56 %
	CE-07	2004.53	3737.00	1949.47	2101.59	50.00 %	94.37 %	1801.52	2004.53	100.00 %	100.00 %
Orig	CE-08	2052.05	4199.00	2083.65	2099.01	84.62 %	97.75 %	1811.40	2054.20	100.00 %	100.00 %
	CE-09	2418.64	5989.00	2448.59	2460.55	76.92 %	97.86 %	2199.23	2432.71	90.91 %	99.29 %
	CE-10	3373.42	7985.00	3359.52	3496.84	77.42 %	96.00 %	3133.53	3394.83	81.48 %	97.18 %
	CE-11	2330.94	8684.00	1929.73	2413.26	70.59 %	95.37 %	2369.02	2332.43	73.33 %	96.33 %
	CE-12	1952.86	4365.00	1806.28	2007.21	45.45 %	93.68 %	1955.47	1966.73	62.50 %	96.84 %
	CE-13	2858.83	8829.00	2448.30	2888.51	64.71 %	94.50 %	2897.97	2891.06	62.50 %	94.55 %
	CE-14	2213.02	4379.00	2082.99	2278.98	83.33 %	96.47 %	2200.13	2215.38	87.50 %	97.67 %
	Average		129.70 %	4.69 %	2.65 %	68.89 %	95.30 %	7.64 %	0.56 %	78.28 %	97.20 %
	CE-01	940.66	982.00	940.61	982.00	75.00 %	50.00 %	830.33	978.90	58.33 %	48.28 %
	CE-02	1605.51	1775.50	1577.03	1671.53	62.79 %	66.67 %	1470.89	1636.13	72.97 %	79.17 %
	CE-03	1706.18	2038.00	1717.45	1756.67	66.67 %	80.56 %	1488.46	1736.72	57.58 %	82.72 %
	CE-04	2237.96	2934.50	2236.85	2307.57	60.00 %	87.30 %	2026.06	2246.90	74.19 %	93.70 %
	CE-05	2770.34	3871.00	2678.19	2850.63	65.85 %	91.86 %	2537.42	2790.77	72.73 %	94.86 %
	CE-06	985.66	1018.00	986.21	1018.00	76.92 %	55.00 %	884.99	998.64	55.56 %	46.67 %
	CE-07	1721.33	1868.50	1701.48	1783.19	80.00 %	71.43 %	1595.98	1764.36	69.57 %	67.44 %
Half-Orig	CE-08	1804.49	2099.50	1816.45	1856.38	58.14 %	76.00 %	1588.91	1840.72	60.53 %	80.52 %
	CE-09	2250.46	2994.50	2253.35	2316.07	60.00 %	87.30 %	2038.56	2259.83	74.19 %	93.70 %
	CE-10	3023.80	3992.50	2940.97	3087.55	66.67 %	89.70 %	2798.92	3045.88	82.61 %	95.03 %
	CE-11	2172.90	4342.00	1749.89	2208.85	91.89 %	96.51 %	2242.91	2218.11	77.78 %	91.30 %
	CE-12	1800.85	2182.50	1641.46	1852.12	50.00 %	83.72 %	1800.17	1809.34	76.92 %	92.50 %
	CE-13	2626.79	4414.50	2199.33	2660.35	86.49 %	94.32 %	2693.73	2661.43	82.35 %	93.48 %
	CE-14	1922.85	2189.50	1773.76	1963.29	66.67 %	87.50 %	1894.48	1985.93	57.89 %	79.49 %
	Average		13.87 %	4.49 %	2.87 %	69.08 %	79.85 %	7.45 %	1.64 %	69.51 %	81.35 %
	CE-01	1082.37	1169.50	1114.50	1127.33	44.44 %	60.53 %	952.56	1133.89	23.53 %	71.74 %
	CE-02	1764.02	2063.50	1738.45	1821.64	40.00 %	86.96 %	1590.29	1781.10	31.25 %	84.29 %
	CE-03	1779.45	2046.00	1822.94	1853.21	41.38 %	80.68 %	1575.03	1829.09	25.00 %	84.21 %
	CE-04	2290.51	2663.50	2356.66	2391.49	43.90 %	82.58 %	2104.98	2345.04	39.29 %	87.77 %
	CE-05	2778.02	3184.75	2769.76	2859.23	57.41 %	86.31 %	2571.70	2851.34	51.11 %	87.50 %
	CE-06	1202.26	1364.00	1241.93	1297.08	37.50 %	77.27 %	1062.18	1245.36	18.18 %	81.25 %
	CE-07	1983.37	2474.50	1957.17	2041.72	50.00 %	94.37 %	1799.22	2005.41	60.00 %	97.22 %
New	CE-08	1948.95	2339.50	2000.75	2006.35	50.00 %	86.36 %	1735.03	1992.19	25.00 %	84.21 %
	CE-09	2310.51	2793.25	2371.39	2355.46	53.33 %	89.55 %	2114.30	2353.67	28.00 %	87.41 %
	CE-10	3147.93	3819.00	3148.71	3234.48	61.54 %	91.43 %	2926.59	3222.40	47.06 %	90.16 %
	CE-11	2203.91	2544.75	1901.63	2289.12	20.45 %	68.47 %	2141.38	2274.49	54.00 %	75.27 %
	CE-12	1894.10	2253.75	1743.71	1994.66	24.00 %	79.79 %	1841.58	1961.81	37.93 %	79.78 %
	CE-13	2857.77	3682.00	2507.60	2922.39	16.67 %	87.18 %	2828.46	2917.94	50.00 %	93.81 %
	CE-14	2083.85	2475.50	1961.03	2193.41	28.57 %	78.26 %	2035.24	2155.48	39.29 %	80.90 %
	Average		28.84 %	4.27 %	3.86 %	40.66 %	82.12 %	7.52 %	2.64 %	37.83 %	84.68 %

Table C.2: Detailed results of the estimation methods: Part 2.

				Goudvis				Huijink			
	Instance	BFS	Prices	EstCosts	AVNS-Fast	Outsourced	Driven	EstCosts	AVNS-Fast	Outsourced	Driven
Orig	CE-01	1119.47	1964.00	1189.68	1119.47	100.00 %	100.00 %	983.04	1130.90	57.14 %	93.48 %
	CE-02	1814.52	3551.00	1898.65	1814.52	100.00 %	100.00 %	1680.52	1814.53	50.00 %	95.83 %
	CE-03	1919.05	4076.00	2071.61	1943.05	60.00 %	93.41 %	1737.97	1930.45	83.33 %	97.78 %
	CE-04	2505.39	5869.00	2608.98	2526.31	76.19 %	96.27 %	2315.16	2523.43	80.00 %	97.01 %
	CE-05	3081.42	7742.00	3175.71	3108.72	78.57 %	96.61 %	2898.67	3098.23	81.48 %	97.18 %
	CE-06	1207.47	2036.00	1276.56	1207.47	100.00 %	100.00 %	1069.91	1217.90	57.14 %	93.48 %
	CE-07	2004.53	3737.00	2087.91	2015.05	50.00 %	95.83 %	1869.23	2004.53	100.00 %	100.00 %
	CE-08	2052.05	4199.00	2211.55	2079.81	71.43 %	95.56 %	1870.77	2066.51	83.33 %	97.78 %
	CE-09	2418.64	5989.00	2529.45	2427.22	75.00 %	97.87 %	2232.14	2428.21	90.91 %	99.29 %
	CE-10	3373.42	7985.00	3468.86	3397.94	78.57 %	96.61 %	3186.47	3389.61	81.48 %	97.18 %
	CE-11	2330.94	8684.00	2335.67	2332.43	73.33 %	96.33 %	2252.01	2363.34	62.50 %	94.55 %
	CE-12	1952.86	4365.00	2009.75	1970.56	62.50 %	96.84 %	1908.88	1969.29	62.50 %	96.84 %
	CE-13	2858.83	8829.00	2864.06	2871.13	73.33 %	96.33 %	2777.41	2878.00	66.67 %	95.45 %
	CE-14	2213.02	4379.00	2250.92	2215.38	87.50 %	97.67 %	2155.29	2215.38	87.50 %	97.67 %
	Average		129.70 %	4.01 %	0.53 %	77.60 %	97.10 %	6.70 %	0.60 %	74.57 %	96.68 %
Half-Orig	CE-01	940.66	982.00	927.88	968.89	76.92 %	55.00 %	839.58	973.77	62.86 %	53.57 %
	CE-02	1605.51	1775.50	1609.83	1662.83	75.56 %	73.17 %	1503.72	1639.39	75.00 %	77.78 %
	CE-03	1706.18	2038.00	1784.90	1773.48	61.70 %	74.65 %	1530.86	1731.33	63.16 %	81.58 %
	CE-04	2237.96	2934.50	2284.70	2306.11	52.94 %	80.49 %	2053.03	2260.08	74.19 %	93.70 %
	CE-05	2770.34	3871.00	2839.66	2815.72	62.50 %	89.35 %	2590.57	2795.46	72.73 %	94.86 %
	CE-06	985.66	1018.00	972.38	1013.39	76.92 %	55.00 %	894.95	1018.00	62.86 %	53.57 %
	CE-07	1721.33	1868.50	1713.53	1790.95	79.25 %	66.67 %	1624.80	1775.58	75.51 %	68.42 %
	CE-08	1804.49	2099.50	1869.52	1859.45	59.57 %	73.61 %	1627.34	1832.81	66.67 %	84.21 %
	CE-09	2250.46	2994.50	2305.20	2299.34	65.00 %	88.71 %	2066.66	2272.58	70.97 %	92.97 %
	CE-10	3023.80	3992.50	3079.71	3076.94	65.52 %	87.58 %	2846.18	3056.64	69.39 %	90.91 %
	CE-11	2172.90	4342.00	2178.94	2186.50	86.11 %	94.38 %	2125.61	2209.51	81.08 %	92.22 %
	CE-12	1800.85	2182.50	1831.35	1860.16	57.14 %	81.25 %	1746.47	1827.52	56.67 %	84.34 %
	CE-13	2626.79	4414.50	2631.24	2640.00	88.57 %	95.51 %	2564.33	2648.95	91.18 %	96.63 %
	CE-14	1922.85	2189.50	1917.50	1981.29	52.38 %	74.36 %	1846.58	2012.64	55.00 %	76.92 %
	Average		28.68 %	1.64 %	2.61 %	68.58 %	77.84 %	6.59 %	1.98 %	69.80 %	81.55 %
New	CE-01	1082.37	1169.50	1130.87	1166.77	46.43 %	59.46 %	936.95	1101.03	43.75 %	79.07 %
	CE-02	1764.02	2063.50	1832.57	1838.66	45.83 %	79.69 %	1594.87	1775.40	46.67 %	88.24 %
	CE-03	1779.45	2046.00	1875.39	1875.27	32.56 %	66.28 %	1576.65	1796.92	40.00 %	86.96 %
	CE-04	2290.51	2663.50	2353.26	2389.51	34.88 %	79.26 %	2082.84	2324.30	46.43 %	89.05 %
	CE-05	2778.02	3184.75	2816.59	2887.52	49.25 %	79.52 %	2560.64	2833.84	48.94 %	86.36 %
	CE-06	1202.26	1364.00	1272.15	1288.59	26.32 %	68.89 %	1049.83	1214.03	66.67 %	93.18 %
	CE-07	1983.37	2474.50	2067.93	1998.23	44.44 %	92.96 %	1816.83	1989.62	80.00 %	98.59 %
	CE-08	1948.95	2339.50	2074.36	2039.92	34.38 %	76.40 %	1740.00	1959.83	42.11 %	88.04 %
	CE-09	2310.51	2793.25	2383.07	2407.07	36.59 %	80.74 %	2093.07	2339.87	30.77 %	87.32 %
	CE-10	3147.93	3819.00	3209.47	3239.35	45.45 %	86.59 %	2917.88	3176.24	44.44 %	89.07 %
	CE-11	2203.91	2544.75	2145.32	2270.55	63.83 %	81.11 %	2073.37	2240.42	65.22 %	82.22 %
	CE-12	1894.10	2253.75	1898.97	1982.44	36.67 %	78.65 %	1798.46	1959.15	36.67 %	78.65 %
	CE-13	2857.77	3682.00	2815.44	2888.20	50.00 %	93.81 %	2722.22	2893.75	50.00 %	93.81 %
	CE-14	2083.85	2475.50	2092.72	2178.19	36.67 %	78.65 %	1991.71	2152.40	36.67 %	78.65 %
	Average		18.11 %	3.17 %	4.19 %	41.66 %	78.71 %	8.57 %	1.46 %	48.45 %	87.09 %

Table C.3: Detailed results of the estimation methods: Part 3.

	Instance	BFS	Prices	BaghuisAdj EstCosts	BaghuisAdj AVNS-Fast	BaghuisAdj Outsourced	BaghuisAdj Driven	Huijink10 EstCosts	Huijink10 AVNS-Fast	Huijink10 Outsourced	Huijink10 Driven
Orig	CE-01	1119.47	1964.00	1150.53	1133.57	62.50 %	93.33 %	1025.34	1133.68	71.43 %	95.56 %
	CE-02	1814.52	3551.00	1741.53	1814.52	100.00 %	100.00 %	1698.88	1814.53	50.00 %	95.83 %
	CE-03	1919.05	4076.00	1936.82	1941.12	71.43 %	95.56 %	1786.81	1932.59	83.33 %	97.78 %
	CE-04	2505.39	5869.00	2524.22	2538.82	77.27 %	96.24 %	2370.33	2521.26	80.00 %	97.01 %
	CE-05	3081.42	7742.00	2999.77	3115.33	89.29 %	98.28 %	2954.19	3097.50	81.48 %	97.18 %
	CE-06	1207.47	2036.00	1238.90	1212.68	71.43 %	95.56 %	1112.34	1222.68	71.43 %	95.56 %
	CE-07	2004.53	3737.00	1931.53	2006.52	50.00 %	95.83 %	1887.59	2004.53	100.00 %	100.00 %
	CE-08	2052.05	4199.00	2073.94	2075.02	91.67 %	98.88 %	1919.71	2061.18	83.33 %	97.78 %
	CE-09	2418.64	5989.00	2429.68	2428.40	75.00 %	97.87 %	2288.84	2432.71	90.91 %	99.29 %
	CE-10	3373.42	7985.00	3283.49	3392.62	81.48 %	97.18 %	3241.99	3389.16	81.48 %	97.18 %
	CE-11	2330.94	8684.00	1911.26	2368.38	68.75 %	95.41 %	2310.57	2406.23	62.50 %	94.55 %
	CE-12	1952.86	4365.00	1766.23	1966.73	62.50 %	96.84 %	1954.12	1969.29	62.50 %	96.84 %
	CE-13	2858.83	8829.00	2435.87	2866.98	62.50 %	94.55 %	2841.98	2867.09	73.33 %	96.33 %
	CE-14	2213.02	4379.00	2047.53	2216.47	87.50 %	97.67 %	2200.30	2215.38	87.50 %	97.67 %
	Average		129.70 %	5.10 %	0.73 %	75.09 %	96.66 %	4.48 %	0.74 %	77.09 %	97.04 %
Half-Orig	CE-01	940.66	982.00	940.52	982.00	70.00 %	0.00 %	862.13	973.77	62.86 %	53.57 %
	CE-02	1605.51	1775.50	1577.11	1666.76	66.67 %	66.67 %	1518.62	1648.23	73.17 %	75.56 %
	CE-03	1706.18	2038.00	1716.19	1746.53	63.41 %	79.73 %	1578.50	1741.14	69.44 %	85.33 %
	CE-04	2237.96	2934.50	2235.82	2277.45	64.86 %	89.68 %	2105.05	2259.61	74.19 %	93.70 %
	CE-05	2770.34	3871.00	2678.71	2827.59	58.70 %	88.95 %	2644.55	2800.43	73.68 %	94.15 %
	CE-06	985.66	1018.00	986.09	1018.00	70.00 %	0.00 %	915.81	1018.00	58.33 %	48.28 %
	CE-07	1721.33	1868.50	1700.78	1777.69	78.00 %	69.44 %	1636.86	1783.36	73.47 %	66.67 %
	CE-08	1804.49	2099.50	1817.19	1843.82	60.98 %	78.67 %	1674.72	1842.31	62.16 %	81.82 %
	CE-09	2250.46	2994.50	2251.61	2286.14	72.22 %	91.94 %	2118.48	2263.35	70.97 %	92.97 %
	CE-10	3023.80	3992.50	2941.75	3076.76	66.00 %	89.76 %	2899.33	3048.30	78.26 %	93.87 %
	CE-11	2172.90	4342.00	1746.27	2191.60	86.49 %	94.32 %	2166.13	2185.01	91.43 %	96.59 %
	CE-12	1800.85	2182.50	1639.76	1825.37	50.00 %	85.06 %	1793.93	1816.68	65.52 %	87.65 %
	CE-13	2626.79	4414.50	2199.75	2638.51	88.57 %	95.51 %	2613.75	2649.95	91.18 %	96.63 %
	CE-14	1922.85	2189.50	1773.06	1954.92	68.97 %	88.75 %	1887.19	1986.66	55.00 %	76.92 %
	Average		28.68 %	4.51 %	2.20 %	68.92 %	72.75 %	4.57 %	1.88 %	71.40 %	81.69 %
New	CE-01	1082.37	1169.50	1113.29	1128.57	43.75 %	50.00 %	962.84	1106.54	43.75 %	79.07 %
	CE-02	1764.02	2063.50	1734.98	1776.36	35.29 %	84.06 %	1615.00	1775.79	46.67 %	88.24 %
	CE-03	1779.45	2046.00	1821.37	1823.81	44.00 %	84.27 %	1616.76	1804.84	40.00 %	86.96 %
	CE-04	2290.51	2663.50	2352.62	2347.59	56.76 %	87.60 %	2134.65	2327.84	46.43 %	89.05 %
	CE-05	2778.02	3184.75	2760.90	2868.86	49.09 %	83.72 %	2603.86	2827.63	58.70 %	88.95 %
	CE-06	1202.26	1364.00	1238.89	1243.49	50.00 %	86.36 %	1075.71	1222.57	50.00 %	88.89 %
	CE-07	1983.37	2474.50	1945.35	1999.26	50.00 %	95.83 %	1836.66	1989.62	80.00 %	98.59 %
	CE-08	1948.95	2339.50	1996.76	1971.97	42.86 %	86.81 %	1780.55	1968.96	35.00 %	86.02 %
	CE-09	2310.51	2793.25	2363.80	2349.42	51.85 %	90.44 %	2149.90	2343.09	46.43 %	89.05 %
	CE-10	3147.93	3819.00	3134.14	3179.81	71.43 %	94.25 %	2964.68	3192.77	51.43 %	90.61 %
	CE-11	2203.91	2544.75	1894.21	2247.97	72.00 %	83.33 %	2109.25	2247.92	61.70 %	80.22 %
	CE-12	1894.10	2253.75	1732.63	1941.05	21.74 %	81.05 %	1835.06	1956.19	36.67 %	78.65 %
	CE-13	2857.77	3682.00	2494.31	2881.48	18.75 %	88.89 %	2777.53	2893.71	50.00 %	93.81 %
	CE-14	2083.85	2475.50	1950.43	2158.20	25.93 %	78.49 %	2028.56	2150.69	36.67 %	78.65 %
	Average		18.11 %	4.43 %	2.16 %	45.25 %	83.94 %	6.71 %	1.67 %	48.82 %	86.91 %

Table C.4: Detailed results of the estimation methods: Part 4.

Instance	BKS	Hall	Baghuis	Fleischmann	Goudvis	Huijink	BaghuisAdj	Huijink10
tai75a	1618.36	2283.30	1233.85	1391.48	1972.47	1445.40	1233.85	1492.15
tai75b	1344.62	1935.84	811.56	1089.94	1719.40	1221.65	811.56	1227.32
tai75c	1291.01	1904.10	1111.16	1085.35	1869.97	1152.34	1111.16	1148.82
tai75d	1365.42	1968.27	814.04	1153.21	1577.30	1164.53	814.04	1186.78
tai100a	2041.34	2827.44	1457.27	1761.47	2515.63	1803.59	1457.27	1813.32
tai100b	1939.90	2751.77	inf	1702.40	2446.35	1706.41	1595.35	1730.10
tai100c	1406.20	2160.55	975.97	1208.68	1874.73	1270.20	975.97	1278.18
tai100d	1581.24	2267.74	1334.10	1319.69	2338.90	1365.23	1334.10	1399.77
tai150a	3055.23	4440.73	1709.39	2772.18	3403.51	2833.03	1709.39	2898.74
tai150b	2656.47	4191.58	inf	2538.21	2965.47	2422.47	1452.88	2476.92
tai150c	2341.84	4143.87	1247.18	2173.43	2648.74	2045.10	1247.18	2067.61
tai150d	2645.39	4061.46	1639.17	2330.79	3207.47	2311.52	1639.17	2354.64
tai385a	24431.44	33258.20	inf	22280.88	33599.67	22218.66	15443.18	22203.17
Avg		47.99	32.14	12.51	25.82	10.98	32.40	9.76

Table C.5: Detailed results of the estimation methods: Part 3.

Appendix D

Parameters and test results of Chapter 5

D.1 Parameters of LNS-Faster

The LNS-Faster has the same moves as the LNS-Fast, with the following exceptions. First, the LNS-Faster has only one move of the create, destroy and tabu search. Second, the following parameters of the tabu search are different.

	Min		Max		NoImpr	
Heuristic	LB	UB	LB	UB	LB	UB
After a move	25	50	200	300	20	40
Tabu search move	50	100	400	600	40	80

Table D.1: The parameters for the tabu search.

Finally, the parameters for the LNS-Faster are presented in Table D.2.

Heuristic	Min	Max	$NoImpr$
LNS-Faster	0	306	51

Table D.2: The parameters for LNS-Faster.

D.2 Detailed test results of the Kriging model

a	b	Costs
3.861	1.949	148,688
3.672	2.320	149,169
3.877	1.662	149,357
3.571	1.913	149,438
1.436	1.879	149,487
2.330	1.775	149,563
6.185	0.197	149,780
1.292	2.770	149,874
8.154	1.383	149,930
0.000	2.274	149,976
0.870	2.009	150,082
2.585	0.554	150,243
2.938	2.095	150,251
2.691	1.350	150,484
4.366	1.780	150,602
7.754	2.215	150,641
4.291	2.375	150,668
6.462	1.108	150,675
1.301	3.600	150,891
7.244	2.758	150,912
5.815	3.323	150,954
0.449	2.867	151,032
5.169	0.000	151,060
1.870	2.294	151,139
1.904	1.149	151,283
5.770	0.722	151,430
3.231	3.877	151,469
0.646	7.200	151,507
6.891	0.252	151,571
1.939	5.538	151,720
3.076	0.000	151,878
8.400	1.800	151,947
0.000	4.430	151,994
6.440	1.819	152,029
4.523	6.092	152,094
8.400	2.880	152,114
7.482	1.374	152,179
0.000	1.645	153,317
8.394	0.763	153,651
7.108	4.985	153,682
0.000	0.000	153,884
8.400	6.646	159,833

Table D.3: Costs without outsourcing.

Company	1	2	3	4	5	6	7	8	9	sum
Day 1	4730.15	7539.29	4221.98	2745.62	3761.8	6075.37	8247	4230.64	2137.83	43689.68
Day 2	4518.08	7825.71	4144.43	3226.43	4028.74	5454.59	8550.15	4105.08	2534.68	44387.89
Day 3	4000.03	6288.47	4324.54	2998.08	3469.99	6634.79	7212.85	4375.76	2631.95	41936.46
Day 4	4554.56	7189.73	4866.67	3558.57	3857.92	5097.1	7466.07	4159.57	2226.12	42976.31
Day 5	5297.11	6612.56	4101.39	2927.54	3790.22	5450.63	7453.19	4955.03	2954.13	43541.8
Day 6	3853.16	6571.27	4093.44	2946.59	4164.22	6062.12	7607.09	4205.44	2607.08	42110.41
Day 7	4130.9	7465.59	4450.6	3156.44	3650.83	5985.52	7350.1	4308.86	2781.29	43280.13
Day 8	4438.85	7110.99	4192.72	3379.76	3671.78	5010.96	7334.23	4419.48	2675.04	42233.81
Day 9	3755.79	8238.82	4551.79	3368.46	4039.63	6198.94	6997.63	4064.78	2646.38	43862.22
Day 10	4310.58	7412.83	4400.14	2899.49	3922.37	4620.61	7150.13	4582.46	2837.9	42136.51
avg	4358.921	7225.526	4334.77	3120.698	3835.75	5659.063	7536.844	4340.71	2603.24	430155.22

Table D.4: Costs without outsourcing.

Appendix E

Details of Chapter 7

E.1 Proof of Theorem 7.2.8

Theorem 7.2.8. *Let* $(E, c) \in BR^N$ *and let* $v_{E,c}$ *be the corresponding bankruptcy game. Then,*

$$\sigma(E, c) = pcn(v_{E,c}).$$

Proof: In this proof, $\sigma(E, c)$ is abbreviated to σ and $v_{E,c}$ is abbreviated to v. Furthermore, since σ and the per capita nucleolus both depend continuously on the estate E, we assume that $\sum_{\ell \in N} \delta_\ell(c) \neq E$.

In order to apply Proposition 6.3.4, we will do the following:
Part I: Define τ, and for all $r \in \{1, ..., \tau\}$, define appropriate relevant collections $\mathcal{D}_r$ and show that $\mathcal{D}_r \subset \mathcal{F}(\bar{\mathcal{D}}_{r-1})$.
Part II: Show that the sequence $\mathcal{D}_1, ..., \mathcal{D}_\tau$ satisfies condition (A) of Proposition 6.3.4, *i.e.*, $\bar{\mathcal{D}}_r = \bigcup_{\ell=1}^r \mathcal{D}_\ell$ is balanced for all $r \in \{1, ..., \tau\}$ and $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$.
Part III: Show that the sequence $\mathcal{D}_1, ..., \mathcal{D}_\tau$ satisfies condition (B) of Proposition 6.3.4, *i.e.*, for all $r \in \{1, ..., \tau\}$ and all $S \in \mathcal{D}_r$, it holds that
$exc^P(v, S, \sigma) = \max_{T \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})} exc^P(v, T, \sigma)$.

Part I: Define

$$t(E, c) := \begin{cases} \min\{\, i \in N \mid \delta_i(c) \geq \alpha\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) > E, \\ \min\{\, i \in N \mid c_i - \delta_i(c) \geq \beta\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) < E, \end{cases} \tag{E.1}$$

in which α and β are determined by $\sum_{\ell \in N} \min\{\alpha, \delta_\ell(c)\} = E$ and $E - \sum_{\ell \in N} \delta_\ell(c) = \sum_{\ell \in N} \max\{0, c_\ell - \delta_\ell(c) - \beta\}$, respectively. For notational convenience, we abbreviate $t(E,c)$ by t. Note that the clights rule allocates the estate in the following way:

$$\sigma_i = \begin{cases} \delta_i(c) & \text{if } i < t, \\ \alpha & \text{if } i \geq t \text{ and } \displaystyle\sum_{\ell \in N} \delta_\ell(c) > E, \\ c_i - \beta & \text{if } i \geq t \text{ and } \displaystyle\sum_{\ell \in N} \delta_\ell(c) < E. \end{cases} \tag{E.2}$$

Additionally, define $S_r := \{1, ..., a_r(c) - 1\} \cup \{r\}$ for each $r \in \{1, ..., t-1\}$ and define

$$\mathcal{D}_r = \{S_r, N \backslash \{r\}\} \text{ for all } r \in \{1, ..., t-1\}.$$

Furthermore, if $t < n$ and $\sum_{\ell \in N} \delta_\ell(c) > E$, then, define $S_r := \{1, ..., m-1\} \cup \{r\}$ for each $r \in \{t, ..., n\}$, where m is defined by

$$m = \max \operatorname*{argmax}_{s \in \{1,...,t\} \,:\, v(\{1,...,s-1\} \cup \{t\}) = 0} \frac{-\alpha - \sum_{\ell=1}^{s-1} \delta_\ell(c)}{s}. \tag{E.3}$$

Such an m exists because $v(\{1, ..., a_t(c) - 1\} \cup \{t\}) = 0$ (see Lemma E.1.1 after this proof, where it is used that $t < n$).

Moreover, if $t < n$, define

$$\mathcal{D}_{(E,c)} = \begin{cases} \{S_t, ..., S_n\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) > E, \\ \{N \backslash \{t\}, ..., N \backslash \{n\}\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) < E. \end{cases}$$

For all $r < t$, we have

$$H(\bar{\mathcal{D}}_r) = \{S : S \subset \{1, ..., r\}\} \cup \{N \backslash S : S \subset \{1, ..., r\}\}.$$

Moreover, if $t < n$, we have

$$H(\bar{\mathcal{D}}_t) = 2^N$$

and if $t = n$, we have

$$H(\bar{\mathcal{D}}_{t-1}) = 2^N,$$

so we define $\tau := \tau(E,c) := \min\{t, n-1\}$.

This gives that $\mathcal{F}(\bar{\mathcal{D}}_{r-1}) = \{S: \ 1 \le |S \cap \{r, ..., n\}| \le n - r\}$ for all $r \le \tau$ which implies that $\mathcal{D}_r \subset \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ for all $r \in \{1, ..., \tau\}$.

Part II: By construction, we have that $\mathcal{F}(\bar{\mathcal{D}}_\tau) = \varnothing$. It remains to prove that $\bar{\mathcal{D}}_r$ is balanced for all $r \in \{1, ..., \tau\}$. The balancedness proof is split into two cases depending on whether $t = 1$ or not.

Case 1: If $t = 1$, then

$$\mathcal{D}_1 = \begin{cases} \{\{1\}, ..., \{n\}\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) > E, \\ \{N\backslash\{1\}, ..., N\backslash\{n\}\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) < E. \end{cases}$$

Choose $\rho(\{i\}) = 1$ for all $i \in N$ to define a balanced map if $\sum_{\ell \in N} \delta_\ell(c) > E$. Choose $\rho(N\backslash\{i\}) = \frac{1}{n-1}$ for all $i \in N$ to define a balanced map if $\sum_{\ell \in N} \delta_\ell(c) < E$.

Case 2: Assume $t > 1$ and let $r \in \{1, ..., \tau\}$. Take $\varepsilon > 0$ sufficiently small. Again, this part is split into two cases. First, we show that $\bar{\mathcal{D}}_r$ is balanced for all $r < t$. Then, we show that $\bar{\mathcal{D}}_t$ is balanced.

Subcase 2a: Let $t > 1$, assume $r < t$ and let

$$\begin{aligned} \rho(S_i) &= \varepsilon \text{ for all } i \in \{2, ..., r\}, \\ \rho(N\backslash\{i\}) &= \varepsilon + \sum_{k=i+1}^{r} \mathbb{1}_{\{i \le a(k)-1\}} \rho(S_k) \text{ for all } i \in \{2, ..., r\}, \\ \rho(N\backslash\{1\}) &= 1 - \sum_{k=2}^{r} \rho(N\backslash\{k\}), \\ \rho(\{1\}) &= \rho(N\backslash\{1\}) - \sum_{k=2}^{r} \rho(S_k). \\ \rho(S) &= 0 \text{ else.} \end{aligned}$$

Since all ρ-values except $\rho(N\backslash\{1\})$ and $\rho(\{1\})$ are in the order of ε, both $\rho(N\backslash\{1\})$ and $\rho(\{1\})$ are strictly positive. Note that $\rho(\{1\}) = \rho(N\backslash\{1\}) = 1$ in case $r = 1$. Let $i > r$. Then,

$$\begin{aligned} \sum_{S: S \ni i} \rho(S) &= \sum_{k=1}^{r} \rho(N\backslash\{k\}) \\ &= \rho(N\backslash\{1\}) + \sum_{k=2}^{r} \rho(N\backslash\{k\}) \\ &= 1 - \sum_{k=2}^{r} \rho(N\backslash\{k\}) + \sum_{k=2}^{r} \rho(N\backslash\{k\}) \end{aligned}$$

$$= 1. \tag{E.4}$$

Let $i \leq r$. Then,

$$\begin{aligned}
\sum_{S:S\ni i} \rho(S) &= \rho(S_i) + \sum_{k=1,k\neq i}^{r} \rho(N\backslash\{k\}) + \sum_{k=i+1}^{r} \mathbb{1}_{\{i\leq a(k)-1\}}\rho(S_k) \\
&= \sum_{k=1}^{r} \rho(N\backslash\{k\}) + \rho(S_i) - \rho(N\backslash\{i\}) + \sum_{k=i+1}^{r} \mathbb{1}_{\{i\leq a(k)-1\}}\rho(S_k) \\
&= \sum_{k=1}^{r} \rho(N\backslash\{k\}) \qquad \text{(E.5)} \\
&= 1, \qquad \text{(E.6)}
\end{aligned}$$

in which equation (E.5) follows from the fact that $\rho(S_i) - \rho(N\backslash\{i\}) = -\sum_{k=i+1}^{r} \mathbb{1}_{\{i\leq a(k)-1\}}\rho(S_k)$ for all $i \leq r$ ($S_1 = \{1\}$ and $1 \in S_k$ for all $k \leq r$). The last equality ((E.6)) is shown in (E.4). Hence, $\sum_{S\in\bar{\mathcal{D}}_r} \rho(S)e^S = e^N$ which completes the proof that this ρ forms a balanced map for the case $r < t$.

Subcase 2b: Let $t > 1$, assume that $r = t$. Since $r \leq \tau = \min\{t, n-1\}$, we have $t = \tau$. Again, this part is split into two, depending on whether $\sum_{\ell\in N} \delta_\ell(c) > E$ or $\sum_{\ell\in N} \delta_\ell(c) < E$.

Subcase 2bi: Let $t > 1$, $r = t = \tau$ and assume $\sum_{\ell\in N} \delta_\ell(c) > E$. Then,

$$\begin{aligned}
\rho(S_i) &= 2\varepsilon \text{ for all } i \in \{2, ..., t-1\}, \\
\rho(S_i) &= \varepsilon \text{ for all } i \in \{t, ..., n\}, \\
\rho(N\backslash\{i\}) &= \varepsilon + \sum_{k=i+1}^{t-1} \mathbb{1}_{\{i\leq a(k)-1\}}\rho(S_k) + \mathbb{1}_{\{i\leq m-1\}} \sum_{k=t}^{n} \rho(S_k) \\
&\qquad\qquad \text{for all } i \in \{2, ..., t-1\}, \\
\rho(N\backslash\{1\}) &= 1 - \varepsilon - \sum_{k=2}^{t-1} \rho(N\backslash\{k\}), \\
\rho(\{1\}) &= \rho(N\backslash\{1\}) + \varepsilon - \sum_{k=2}^{t-1} \mathbb{1}_{\{1\leq a(k)-1\}}\rho(S_k) - \mathbb{1}_{\{1\leq m-1\}} \sum_{k=t}^{n} \rho(S_k). \\
\rho(S) &= 0 \text{ else.}
\end{aligned}$$

Since all ρ-values except $\rho(N\backslash\{1\})$ and $\rho(\{1\})$ are in the order of ε, both $\rho(N\backslash\{1\})$ and $\rho(\{1\})$ are strictly positive.

Let $i \geq t$. Then,

$$\begin{aligned}
\sum_{S:S\ni i} \rho(S) &= \rho(S_i) + \sum_{k=1}^{t-1} \rho(N\backslash\{k\}) \\
&= \varepsilon + \rho(N\backslash\{1\}) + \sum_{k=2}^{t-1} \rho(N\backslash\{k\}) \\
&= \varepsilon + 1 - \varepsilon - \sum_{k=2}^{t-1} \rho(N\backslash\{k\}) + \sum_{k=2}^{t-1} \rho(N\backslash\{k\}) \\
&= 1.
\end{aligned}$$

Let $i < t$. Then,

$$\begin{aligned}
\sum_{S:S\ni i} \rho(S) &= \rho(S_i) + \sum_{k=1,k\neq i}^{t-1} \rho(N\backslash\{k\}) + \mathbb{1}_{\{i\leq m-1\}} \sum_{k=t}^{n} \rho(S_k) \\
&\qquad + \sum_{k=i+1}^{t-1} \mathbb{1}_{\{i\leq a(k)-1\}} \rho(S_k) \\
&= \sum_{k=1}^{t-1} \rho(N\backslash\{k\}) + \rho(S_i) - \rho(N\backslash\{i\}) + \mathbb{1}_{\{i\leq m-1\}} \sum_{k=t}^{n} \rho(S_k) \\
&\qquad + \sum_{k=i+1}^{t-1} \mathbb{1}_{\{i\leq a(k)-1\}} \rho(S_k) \\
&= \sum_{k=1}^{t-1} \rho(N\backslash\{k\}) + \varepsilon \qquad \text{(E.7)} \\
&= \sum_{k=2}^{t-1} \rho(N\backslash\{k\}) + \rho(N\backslash\{1\}) + \varepsilon \\
&= (1-\varepsilon) + \varepsilon \\
&= 1.
\end{aligned}$$

Again, (E.7) follows from the fact that $\rho(S_i) - \rho(N\backslash\{i\}) = \varepsilon - \mathbb{1}_{\{i\leq m-1\}} \sum_{k=t}^{n} \rho(S_k) \sum_{k=i+1}^{t-1} \mathbb{1}_{\{i\leq a(k)-1\}} \rho(S_k)$. Hence, $\sum_{S\in\bar{\mathcal{D}}_t} \rho(S)e^S = e^N$, which completes Case 2bi.

Subcase 2bii: Let $t > 1$, $r = t = \tau$, and assume $\sum_{\ell\in N} \delta_\ell(c) < E$. Then, $\bar{\mathcal{D}}_t = \bar{\mathcal{D}}_{t-1} \cup \mathcal{B}$, where $\mathcal{B} = \{N\backslash\{1\}, ..., N\backslash\{n\}\}$. Since $\mathcal{B}$ is balanced (see Case 1) and $\bar{\mathcal{D}}_{t-1}$ is balanced (see Subcase 2a), we have that $\bar{\mathcal{D}}_t$ is balanced.

Part II: For all cases it is shown that $\bar{\mathcal{D}}_r$ is balanced for all $r \in \{1, ..., \tau\}$.

Part III: The proof is split into two parts. In the first part (Part IIIA), we provide an upper bound for the per capita excesses of coalitions in $\mathcal{F}(\bar{\mathcal{D}}_{r-1})$. In the second part (Part IIIB), it is shown that the coalitions in $S \in \mathcal{D}_r$ are equal to this upper bound.

First, note that $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ implies that there exists at least one $j \geq r$ such that $j \in S$.

Part IIIA: Let $r \in \{1, ..., \tau\}$ and let $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$. The proof is split into two cases, depending on whether $v(S) = 0$ or $v(S) > 0$.

Case 1: Let $r \in \{1, ..., \tau\}$ and let $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$. Assume that $v(S) = 0$ and define $s = \min\{|S|, r\}$. Again there are two cases, depending on whether $\delta_r(c) \leq \sigma_r$ or $\delta_r(c) > \sigma_r$.

Subcase 1i: Let $r \in \{1, ..., \tau\}$, $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ and let $v(S) = 0$. Assume that $\delta_r(c) \leq \sigma_r$. We have

$$\begin{aligned}
exc^P(v, S, \sigma) &= \frac{0 - \sum_{\ell \in S} \sigma_\ell}{|S|} \\
&\leq \frac{0 - \sum_{\ell=1}^{s-1} \sigma_\ell - \sigma_r}{s} && \text{(E.8)} \\
&\leq \frac{0 - \sum_{\ell=1}^{s-1} \delta_\ell(c) - \delta_r(c)}{s} && \text{(E.9)} \\
&= \frac{-\sum_{\ell=1}^{s-1} \delta_\ell(c) - \frac{n+s-1}{n-1}\delta_r(c) + \frac{s}{n-1}\delta_r(c)}{s} \\
&= \frac{-\sum_{\ell=1}^{s-1} \delta_\ell(c) - \frac{n+s-1}{n-1}\delta_r(c)}{s} + \frac{\delta_r(c)}{n-1} \\
&\leq \frac{-\sum_{\ell=1}^{s-1} \delta_\ell(c) - \frac{1}{n-1}(sc_r - (n-1)\sum_{\ell=1}^{s-1} \delta_\ell(c))}{s} + \frac{\delta_r(c)}{n-1} && \text{(E.10)} \\
&= \frac{\delta_r(c) - c_r}{n-1}.
\end{aligned}$$

To clarify, at (E.8) the following is used: If $r > s$, then, we first remove the $r - s$ players which receive the most. Then, we replace the remaining players with players $1, ..., s-1$ and r, which is possible since there is a player $j \in S$ such that $j \geq r$. At (E.9), it is used that $\delta_r(c) \leq \sigma_r$, which implies that $\delta_\ell(c) \leq \sigma_\ell$ for all $\ell \in \{1, ..., r\}$. Finally, (E.10) follows from (7.2).

To conclude this subcase: for all $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ with $v(S) = 0$ and $\delta_r(c) \leq \sigma_r$, we have that $exc^P(v, S, \sigma) \leq \frac{\delta_r(c) - c_r}{n-1}$.

Subcase 1ii: Let $r \in \{1, ..., \tau\}$, let $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$, and let $v(S) = 0$. Assume that $\delta_r(c) > \sigma_r$. This implies due to the definition of the clights rule, Definition 7.2.1, that $\sum_{\ell \in N} \delta_\ell > E$ and $r \geq t$. Then, since $r \in \{1, ..., \tau\}$, the definition of τ and $r \geq t$ we have that $r = t$. Furthermore, we have

$$\begin{aligned}
exc^P(v, S, \sigma) &\leq \frac{0 - \sum_{\ell=1}^{s-1} \sigma_\ell - \sigma_r}{s} && \text{(E.11)}\\
&= \frac{0 - \sum_{\ell=1}^{s-1} \sigma_\ell - \sigma_t}{s}\\
&= \frac{-\sum_{\ell=1}^{s-1} \delta_\ell(c) - \alpha}{s} && \text{(E.12)}\\
&\leq \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m}. && \text{(E.13)}
\end{aligned}$$

To clarify, (E.11) follows from Subcase 1i until (E.8). At (E.12) we use (E.2) and at (E.13) we use (E.3) together with the fact that $v(S) = 0$ implies that $v(\{1, ..., s-1\} \cup \{r\}) = 0$.

To conclude this subcase: if $\delta_t(c) > \sigma_t$ (and hence $\sum_{\ell \in N} \delta_\ell > E$), we have for all $S \in \mathcal{F}(\bar{\mathcal{D}}_{t-1})$ with $v(S) = 0$ that $exc^P(v, S, \sigma) \leq \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m}$.

Subcase 1: From Subcase 1i and Subcase 1ii, we obtain that for all
$r \in \{1, ..., \tau\}$ and all $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ with $v(S) = 0$,

$$exc^P(v, S, \sigma) \leq \begin{cases} \frac{\delta_r(c) - c_r}{n-1} & \text{if } r < t,\\ \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} & \text{if } r = t \text{ and } \sum_{\ell \in N} \delta_\ell(c) > E,\\ \frac{\delta_t(c) - c_t}{n-1} & \text{if } r = t \text{ and } \sum_{\ell \in N} \delta_\ell(c) < E. \end{cases}$$

Subcase 2: Let $r \in \{1, ..., \tau\}$, let $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$, assume that $v(S) > 0$ and let $k \in \{r, ..., n\} \backslash S$. We have

$$\begin{aligned}
exc^P(v, S, \sigma) &= \frac{v(S) - \sum_{\ell \in S} \sigma_\ell}{|S|}\\
&\leq \frac{v(S) - \sum_{\ell \in S} \sigma_\ell + \sum_{\ell \in N \backslash (S \cup \{k\})} c_\ell - \sum_{\ell \in N \backslash (S \cup \{k\})} \sigma_\ell}{|S|} && \text{(E.14)}\\
&= \frac{E - \sum_{\ell \in N \backslash S} c_\ell - \sum_{\ell \in S} \sigma_\ell + \sum_{\ell \in N \backslash (S \cup \{k\})} c_\ell - \sum_{\ell \in N \backslash (S \cup \{k\})} \sigma_\ell}{|S|}\\
&= \frac{E - c_k - (E - \sigma_k)}{|S|}
\end{aligned}$$

$$
\begin{aligned}
&\leq \frac{\sigma_k - c_k}{n-1} \\
&\leq \frac{\sigma_r - c_r}{n-1}. \qquad \text{(E.15)}
\end{aligned}
$$

To clarify, (E.14) uses the fact that $\sigma_i \leq c_i$ for all $i \in N$. Inequality (E.15) follows from the fact that $r \leq k$ together with the fact that $c_{i-1} - \sigma_{i-1} \leq c_i - \sigma_i$ for all $i \in \{2, ..., n\}$ (Lemma 7.2.7).

To conclude this subcase: For all $r \in \{1, ..., \tau\}$ and all $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ with $v(S) > 0$, we have, due to (E.2), that

$$
exc^P(v, S, \sigma) \leq \frac{\sigma_r - c_r}{n-1} = \begin{cases} \frac{\delta_r(c) - c_r}{n-1} & \text{if } r < t, \\ \frac{\alpha - c_r}{n-1} & \text{if } r = t \text{ and } \sum_{\ell \in N} \delta_\ell(c) > E, \\ \frac{-\beta}{n-1} & \text{if } r = t \text{ and } \sum_{\ell \in N} \delta_\ell(c) < E. \end{cases}
$$

Part IIIA: We have from Subcase 1 and Subcase 2 that for all $r \in \{1, ..., \tau\}$ and all $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$, it holds that

$$
exc^P(v, S, \sigma) \leq \begin{cases} \dfrac{\delta_r(c) - c_r}{n-1} & \text{if } r < t, \\ \dfrac{-\sum\limits_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} & \text{if } r = t, \sum\limits_{\ell \in N} \delta_\ell(c) > E \text{ and } v(S) = 0, \\ \dfrac{\alpha - c_t}{n-1} & \text{if } r = t, \sum\limits_{\ell \in N} \delta_\ell(c) > E \text{ and } v(S) > 0, \\ \dfrac{\delta_t - c_t}{n-1} & \text{if } r = t, \sum\limits_{\ell \in N} \delta_\ell(c) < E \text{ and } v(S) = 0, \\ \dfrac{-\beta}{n-1} & \text{if } r = t, \sum\limits_{\ell \in N} \delta_\ell(c) < E \text{ and } v(S) > 0. \end{cases}
$$

Now, we will show for the case $r = t$, that $\frac{a - c_t}{n-1} \leq \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m}$ and $\frac{\delta_t - c_t}{n-1} \leq \frac{-\beta}{n-1}$. If $r = t$ and $\sum_{\ell \in N} \delta_\ell(c) > E$, then

$$
\begin{aligned}
\frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} &\geq \frac{-\sum_{\ell=1}^{a_t(c)-1} \delta_\ell(c) - \alpha}{a_t(c)} && \text{(E.16)} \\
&= \frac{-\left(\frac{a_t(c)}{n-1} c_t - \frac{n + a_t(c) - 1}{n-1} \delta_t(c)\right) - \alpha}{a_t(c)} && \text{(E.17)} \\
&\geq \frac{-\frac{a_t(c)}{n-1} c_t + \frac{n + a_t(c) - 1}{n-1} \alpha - \alpha}{a_t(c)} && \text{(E.18)}
\end{aligned}
$$

$$= \frac{\alpha - c_t}{n-1}.$$

To clarify, (E.16) uses (E.3), which also uses Lemma E.1.1, and (E.17) uses (7.3). Finally, at (E.18), we use (E.1).

Furthermore, for the case $r = t$ and $\sum_{\ell \in N} \delta_\ell(c) < E$ we have by (E.1) that $-\beta \geq \delta_t(c) - c_t$ which implies that $\frac{\delta_t - c_t}{n-1} \leq \frac{-\beta}{n-1}$. Hence, we obtain for all $r \in \{1, ..., \tau\}$ and all $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$ the following:

$$exc^P(v, S, \sigma) \leq \begin{cases} \dfrac{\delta_r(c) - c_r}{n-1} & \text{if } r < t, \\ \dfrac{-\sum\limits_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} & \text{if } r = t \text{ and } \sum\limits_{\ell \in N} \delta_\ell(c) > E, \\ \dfrac{-\beta}{n-1} & \text{if } r = t \text{ and } \sum\limits_{\ell \in N} \delta_\ell(c) < E. \end{cases} \tag{E.19}$$

Part IIIB: Let $r \in \{1, ..., \tau\}$. The proof is split into two cases, depending on whether $r < t$ or not.

Subcase 1: Let $r \in \{1, ..., \tau\}$ and assume that $r < t$. Then, $\mathcal{D}_r = \{S_r, N \backslash \{r\}\}$ and we have for S_r, that

$$exc^P(v, S_r, \sigma) = \frac{0 - \sum_{\ell=1}^{a_r(c)-1} \sigma_\ell - \sigma_r}{a_r(c)} \tag{E.20}$$

$$= \frac{0 - \sum_{\ell=1}^{a_r(c)-1} \delta_\ell - \delta_r}{a_r(c)} \tag{E.21}$$

$$= \frac{-\sum\limits_{\ell=1}^{a_r(c)-1} \delta_\ell(c) - \dfrac{n + a_r(c) - 1}{n-1}\delta_r(c) + \dfrac{a_r(c)}{n-1}\delta_r(c)}{a_r(c)}$$

$$= \frac{-\sum\limits_{\ell=1}^{a_r(c)-1} \delta_\ell(c) - \dfrac{1}{n-1}(a_r(c)c_r - (n-1)\sum\limits_{\ell=1}^{a_r(c)-1} \delta_\ell(c))}{a_r(c)} + \frac{\delta_r(c)}{n-1} \tag{E.22}$$

$$= \frac{\delta_r(c) - c_r}{n-1}.$$

To clarify, at (E.20), (E.21), and (E.22), we use Lemma E.1.1, (E.2), and (7.3), respectively.

For $N\backslash\{r\}$, we have

$$exc^P(v, N\backslash\{r\}, \sigma) = \frac{v(N\backslash\{r\}) - \sum_{\ell\in N\backslash\{r\}} \sigma_\ell}{n-1}$$
$$= \frac{E - c_r - (E - \sigma_r)}{n-1} \tag{E.23}$$
$$= \frac{\delta_r(c) - c_r}{n-1}. \tag{E.24}$$

To clarify, at (E.23) we use Lemma E.1.2 and at (E.24) we use (E.2).

To conclude this subcase: $exc^P(v, S_r, \sigma) = exc^P(v, N\backslash\{r\}, \sigma) = \frac{\delta_r(c) - c_r}{n-1}$.

Subcase 2: Let $r \in \{1, ..., \tau\}$ and assume that $r = t$. There are two cases, depending on whether $\sum_{\ell\in N} \delta_\ell(c) > E$ or $\sum_{\ell\in N} \delta_\ell(c) < E$.
Subcase 2i: Let $r \in \{1, ..., \tau\}$, let $r = t$, and assume that $\sum_{\ell\in N} \delta_\ell(c) > E$. Then, $\mathcal{D}_t = \{S_t, ..., S_n\}$ and, by (E.3) and the construction of S_t, we have that $v(S_t) = 0$ and

$$exc^P(v, S_t, \sigma) = \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m}.$$

Furthermore, for S_j with $j > t$ we have that

$$\frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} \geq exc^P(v, S_j, \sigma) \tag{E.25}$$
$$\geq \frac{0 - \sum_{\ell\in S_j} \sigma_\ell}{|S_j|} \tag{E.26}$$
$$= \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m}. \tag{E.27}$$

To clarify: (E.25), (E.26) and (E.27), follow from (E.19), the fact that $v(S_j) \geq 0$, and the definition of S_j, respectively.

To conclude this subcase: $exc^P(v, T, \sigma) = \frac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m}$ for all $T \in \mathcal{D}_t$.

Subcase 2ii: Let $r = t$ and assume that $\sum_{\ell\in N} \delta_\ell(c) < E$. Then,
$\mathcal{D}_t = \{N\backslash\{t\}, ..., N\backslash\{n\}\}$ and for all $j \geq t$, we have

$$exc^P(v, N\backslash\{j\}, \sigma) = \frac{v(N\backslash\{j\}) - \sum_{\ell\in N\backslash\{j\}} \sigma_\ell}{n-1}$$

$$= \frac{E - c_j - (E - \sigma_j)}{n-1} \tag{E.28}$$
$$= \frac{-\beta}{n-1}. \tag{E.29}$$

To clarify, at (E.28), we use Lemma E.1.2 and at (E.29), we use $r = t$ together with (E.2).

To conclude this subcase: $exc^P(v, T, \sigma) = \frac{-\beta}{n-1}$ for all $T \in \mathcal{D}_t$.

Part IIIB: From Subcase 1, Subcase 2i, and Subcase 2ii, we obtain for all $r \in \{1, ..., \tau\}$ and all $S \in \mathcal{D}_r$, that

$$exc^P(v, S, \sigma) = \begin{cases} \dfrac{\delta_r(c) - c_r}{n-1} & \text{if } r < t, \\ \dfrac{-\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} & \text{if } r = t \text{ and } \displaystyle\sum_{\ell \in N} \delta_\ell(c) > E, \\ \dfrac{-\beta}{n-1} & \text{if } r = t \text{ and } \displaystyle\sum_{\ell \in N} \delta_\ell(c) < E. \end{cases} \tag{E.30}$$

Part III: From **Part IIIA**, especially (E.19), and from **Part IIIB**, especially (E.30), we obtain for all $r \in \{1, ..., \tau\}$, all $S \in \mathcal{F}(\bar{\mathcal{D}}_{r-1})$, and all $T \in \mathcal{D}_r$,

$$exc^P(v, S, \sigma) \leq exc^P(v, T, \sigma) = \begin{cases} \dfrac{\delta_r(c) - c_r}{n-1} & \text{if } r < t, \\ \dfrac{-\displaystyle\sum_{\ell=1}^{m-1} \delta_\ell(c) - \alpha}{m} & \text{if } r = t \text{ and } \displaystyle\sum_{\ell \in N} \delta_\ell(c) > E, \\ \dfrac{-\beta}{n-1} & \text{if } r = t \text{ and } \displaystyle\sum_{\ell \in N} \delta_\ell(c) < E. \end{cases}$$

Hence, by Proposition 6.3.4, we obtain $\sigma(E, c) = pcn(v_{E,c})$. □

Lemma E.1.1 *Let $(E, c) \in BR^N$ be a problem and let $t(E, c)$ as defined in* (E.1). *Let $t(E, c) < n$. Then, for all*

$$i \in \begin{cases} \{1, \ldots, n-1\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) > E, \\ \{1, \ldots, t(E,c) - 1\} & \text{if } \sum_{\ell \in N} \delta_\ell(c) < E, \end{cases}$$

it holds that

$$v_{E,c}(\{1,\dots,a_i(c)-1\}\cup\{i\})=0.$$

Proof: For notational convenience, we abbreviate $t(E,c)$ to t. Let $k=n-1$ if $\sum_{\ell\in N}\delta_\ell(c)>E$, and let $k=t-1$ if $\sum_{\ell\in N}\delta_\ell(c)<E$. Proving that $v_{E,c}(\{1,...,a_i(c)-1\}\cup\{i\})=0$ for $i=k$ implies that it holds for all $i\in\{1,...,k\}$. The worth of the coalition is given by

$$v_{E,c}(\{1,...,a_k(c)-1\}\cup\{k\})=\max\{0,E-\sum_{i=a_k(c)}^{k-1}c_i-\sum_{i=k+1}^{n}c_i\},$$

hence, proving that $E-\sum_{i=a_k(c)}^{k-1}c_i-\sum_{i=k+1}^{n}c_i\le 0$ is sufficient.
First, we will prove that

$$E\le\sum_{i=1}^{k}\delta_i(c)+\sum_{i=k+1}^{n}c_i-(n-k)(c_k-\delta_k(c)).\tag{E.31}$$

If $\sum_{\ell\in N}\delta_\ell(c)>E$, so $k=n-1$, we have

$$\begin{aligned}E&\le\sum_{i\in N}\delta_i(c)\\&=\sum_{i=1}^{n-1}\delta_i(c)+\delta_n(c)\\&\le\sum_{i=1}^{n-1}\delta_i(c)+c_n-c_{n-1}+\delta_{n-1}(c),\end{aligned}\tag{E.32}$$

where (E.32) follows from Lemma 7.2.7.
If $\sum_{\ell\in N}\delta_\ell(c)<E$, so $k=t-1$, we have

$$\begin{aligned}E&=\sum_{i=1}^{t-1}\delta_i(c)+\sum_{i=t}^{n}(c_i-\beta)\\&=\sum_{i=1}^{t-1}\delta_i(c)+\sum_{i=t}^{n}c_i-(n-t+1)\beta\\&<\sum_{i=1}^{t-1}\delta_i(c)+\sum_{i=t}^{n}c_i-(n-t+1)(c_{t-1}-\delta_{t-1}(c)),\end{aligned}\tag{E.33}$$

where (E.33) follows from (E.1).

Now, we will prove that $E - \sum_{i=a_k(c)}^{k-1} c_i - \sum_{i=k+1}^{n} c_i \leq 0$:

$$
\begin{aligned}
& E - \sum_{i=a_k(c)}^{k-1} c_i - \sum_{i=k+1}^{n} c_i \\
\leq\ & \sum_{i=1}^{k} \delta_i(c) + \sum_{i=k+1}^{n} c_i - (n-k)(c_k - \delta_k(c)) - \sum_{i=a_k(c)}^{k-1} c_i - \sum_{i=k+1}^{n} c_i && \text{(E.34)} \\
=\ & \sum_{i=1}^{a_k(c)-1} \delta_i(c) + \sum_{i=a_k(c)}^{k-1} \delta_i(c) + \delta_k(c) + (n-k)\delta_k(c) + (k-n)c_k - \sum_{i=a_k(c)}^{k-1} c_i \\
=\ & \sum_{i=1}^{a_k(c)-1} \delta_i(c) + (n-k+1)\delta_k(c) + (k-n)c_k - \sum_{i=a_k(c)}^{k-1} (c_i - \delta_i(c)) \\
\leq\ & \sum_{i=1}^{a_k(c)-1} \delta_i(c) + (n-k+1)\delta_k(c) + (k-n)c_k \\
=\ & \sum_{i=1}^{a_k(c)-1} \delta_i(c) + (n-k+1)\Big(\frac{a_k(c)}{n+a_k(c)-1} c_k - \frac{n-1}{n+a_k(c)-1} \sum_{i=1}^{a_k(c)-1} \delta_i(c)\Big) + (k-n)c_k && \text{(E.35)} \\
=\ & \Big(\frac{(n-k+1)a_k(c)}{n+a_k(c)-1} + (k-n)\Big)c_k + \Big(\frac{(n-k+1)(1-n)}{n+a_k(c)-1} + 1\Big) \sum_{i=1}^{a_k(c)-1} \delta_i(c) \\
=\ & \Big(\frac{a_k(c) + (k-n)(n-1)}{n+a_k(c)-1}\Big)c_k + \Big(\frac{a_k(c) + (k-n)(n-1)}{n+a_k(c)-1}\Big) \sum_{i=1}^{a_k(c)-1} \delta_i(c) \\
\leq\ & 0. && \text{(E.36)}
\end{aligned}
$$

To clarify, at (E.34), (E.35), and (E.36), we have used (E.31), (7.3), and the fact that $a_k(c) \leq k < n$, respectively. □

Lemma E.1.2 *Let* $(E, c) \in BR^N$ *be a problem with* $t(E, c)$ *as* (E.1). *Then, for all*

$$
i \in \begin{cases} \{1, \ldots, t(E,c) - 1\} & \textit{if } \sum_{\ell \in N} \delta_\ell(c) > E, \\ \{1, \ldots, n(E,c) - 1\} & \textit{if } \sum_{\ell \in N} \delta_\ell(c) < E \textit{ and } t(E,c) = n, \\ \{1, \ldots, n\} & \textit{if } \sum_{\ell \in N} \delta_\ell(c) < E \textit{ and } t(E,c) < n, \end{cases}
$$

it holds that

$$
E - c_i \geq 0
$$

Proof: For notational convenience, we abbreviate $t(E,c)$ to t. Note that proving $E \geq c_i$ for $i = t-1$ (or $i = n-1$ or $i = n$) implies that it holds for all $i \in \{1, ..., t-1\}$ (or $i \in \{1, ..., n-1\}$ or $i \in N$). The proof is split into two cases, depending on whether $\sum_{\ell \in N} \delta_\ell(c) > E$ or $\sum_{\ell \in N} \delta_\ell(c) < E$.

Case 1: Assume $\sum_{\ell \in N} \delta_\ell(c) > E$. Then, by (E.1), we have

$$E > \sum_{\ell=1}^{t-1} \delta_\ell(c) + \sum_{\ell=t}^{n} \delta_{t-1}(c). \tag{E.37}$$

By (7.2), we have that

$$\delta_{t-1}(c) \geq \frac{t-1}{n+t-1-1} c_{t-1} - \frac{n-1}{n+t-1-1} \sum_{\ell=1}^{t-1-1} \delta_\ell(c),$$

which can be rewritten to

$$\sum_{\ell=1}^{t-2} \delta_\ell(c) \geq \frac{t-1}{n-1} c_{t-1} - \frac{n+t-2}{n-1} \delta_{t-1}(c). \tag{E.38}$$

Now, we will prove that $E - c_{t-1} \geq 0$:

$$\begin{aligned}
E - c_{t-1} &> \sum_{\ell=1}^{t-2} \delta_\ell(c) + (n-t+2)\delta_{t-1}(c) - c_{t-1} && \text{(E.39)} \\
&\geq \frac{t-1}{n-1} c_{t-1} - \frac{n+t-2}{n-1} \delta_{t-1}(c) + (n-t+2)\delta_{t-1}(c) + \frac{1-n}{n-1} c_{t-1} && \text{(E.40)} \\
&= \frac{t-n}{n-1} c_{t-1} - \frac{n+t-2}{n-1} \delta_{t-1}(c) + \frac{n^2 - nt + 2n - n + t - 2}{n-1} \delta_{t-1}(c) \\
&= \frac{t-n}{n-1} c_{t-1} + \frac{n^2 - nt}{n-1} \delta_{t-1}(c) \\
&= \frac{n-t}{n-1} (n\delta_{t-1}(c) - c_{t-1}) \\
&\geq 0. && \text{(E.41)}
\end{aligned}$$

To clarify, at (E.39), (E.40), and (E.41), we use (E.37), (E.38), and Lemma 7.2.2, respectively.

Case 2: Assume $\sum_{\ell \in N} \delta_\ell(c) < E$. Again, this case is split into two parts, depending whether $t = n$ or $t < n$.

Case 2a: We have $\sum_{\ell \in N} \delta_\ell(c) < E$ and assume that $t = n$. Then,

$$E - c_{n-1} > \sum_{\ell \in N} \delta_\ell(c) - c_{n-1}$$

$$\geq \sum_{j=1}^{n-2} \delta_j(c) + 2\delta_{n-1}(c) - c_{n-1} \tag{E.42}$$

$$\geq \sum_{j=1}^{n-2} \delta_j(c) + 2\Big(\frac{1}{2}c_{n-1} - \frac{1}{2}\sum_{j=1}^{n-2} \delta_j(c)\Big) - c_{n-1} \tag{E.43}$$

$$= 0.$$

To clarify, at (E.42) we use Lemma 7.2.3 and at (E.43) we use (7.2).

Case 2b: We have $\sum_{\ell \in N} \delta_\ell(c) < E$ and assume that $t < n$. Then,

$$\begin{aligned} E &= \sum_{\ell=1}^{t-1} \delta_\ell(c) + \sum_{\ell=t}^{n} (c_\ell - \beta) \\ &= \sum_{\ell=1}^{t-1} \delta_\ell(c) + c_n - \beta + \sum_{\ell=t}^{n-1} (c_\ell - \beta) \\ &\geq \sum_{\ell=1}^{t-1} \delta_\ell(c) + c_n - (c_{n-1} - \delta_{n-1}(c)) + \sum_{\ell=t}^{n-1} (c_\ell - (c_\ell - \delta_\ell(c))) && \text{(E.44)} \\ &= \sum_{j=1}^{n-2} \delta_j(c) + 2\delta_{n-1}(c) + c_n - c_{n-1}. \end{aligned}$$

To clarify, at (E.44) we use (E.1) together with the fact that $c_{i-1} - \sigma_{i-1} \leq c_i - \sigma_i$ for all $i \in \{2, ..., n\}$ (Lemma 7.2.7). Using this, we have

$$\begin{aligned} E - c_n &\geq \sum_{j=1}^{n-2} \delta_j(c) + 2\delta_{n-1}(c) + c_n - c_{n-1} - c_n \\ &= \sum_{j=1}^{n-2} \delta_j(c) + 2\delta_{n-1}(c) - c_{n-1} \\ &\geq 0. && \text{(E.45)} \end{aligned}$$

To clarify, at (E.45) we use Case 2a. $\square$

Glossary

Abbreviations of routing problems	
CPTP	Capacitated profitable tour problem
TOP	Team orienteering problem
VRP	Vehicle routing problem
VRPO	Vehicle routing problem with order outsourcing
VRPPC	Vehicle routing problem with private fleet and common carrier

Order characteristics	
$\{0, 1, ..., n\}$	The set of orders, where 0 denotes the depot.
p_i	Outsource costs for order i.
q_i	Demand of order i.
d_{ij}	The distance from the location of order i to the location of order j.

Truck (route) characteristics	
m	The number of available trucks.
Q	The capacity of a truck.
F	The fixed costs of a truck.
v	The additional costs per unit distance of a truck.
$infeasibility$	The amount of demand more than the capacity of a truck squared.

Moves in Chapter 3	
driven routes	A route with less delivery costs than outsource costs.
candidate routes	A route with more delivery costs than outsource costs.
delivered orders	Orders that are in driven routes.
outsourced orders	Orders that are in candidate routes or not in any route.
insertion costs	The additional distance costs for delivering an order in a route.
profitable insertion	A variant of the classical insertion method which takes the outsource costs into account.
profit (of an order)	Outsource costs minus the insertion costs divided by the demand.
1-shift	Moves one order from one route to another one.
1-swap	Switches two orders from different routes.
tabu	Forbids a move.
remove-and-insert move	Randomly outsources delivered orders and randomly delivers outsourced orders.
cyclic move	Cyclicly moves orders from one route to another one.
shifting procedure	Repairs a solution by using profitable insertion which also allows insertions from delivered routes.
create move	Creates a route with one outsourced order. Followed by the shifting procedure to fill the new route with orders.
destroy move	Outsources all orders in a route.
split move	Splits a driven route into two. Followed by the shifting procedure in order to fill these routes.
bomb move	Ruins a solution by outsourcing orders with the location closest to the center. Followed by the shifting procedure.
reseeding	Ruins a solution by outsourcing parts of routes. Followed by the shifting procedure.

Abbreviations of the heuristics	
AVNS	adaptable variable neighborhood search (Stenger, Vigo, Enz, and Schwind (2013))
AVNS-RN	adaptable variable neighborhood search-random neighborhood ordering (Stenger, Schneider, and Goeke (2013))
GA	genetic algorithm (Kratica et al. (2012))
LNS-Fast (and Slow)	large neighborhood search-Fast (Slow) (Huijink, Kant, and Peeters (2014))
MS-ILS	multistart iterative local search (Vidal et al. (2015))
MS-LS	multistart local-improvement (Vidal et al. (2015))
R-TS	reseeding-tabu search (Huijink, Kant, and Peeters (2014))
RIP	randomized construction-improvement-perturbation (Bolduc et al. (2008))
TS(25)	tabu search (Côté and Potvin (2009))
TS	tabu search (Potvin and Naud (2011))
TS+	tabu search with ejection chains (Potvin and Naud (2011))
UHGS	unified hybrid genetic search (Vidal et al. (2015))

Symbols used in Chapter 4	
$EstCosts_i$	Estimated delivery costs of order i.
F_i	Estimated fixed costs of order i.
S_i	Estimated stem costs of order i.
IO_i	Estimated inter-order costs of order i.
s_i	Score of order i.
$\pi(i)$	Rank of order i.
$Costs$	The true costs of an estimation.
$BKSCosts$	The costs of the best known solution.
$Both$	The set of orders that are outsourced (delivered) by both the estimation and the best known solution.
$Either$	The set of orders that are outsourced (delivered) by either the estimation or the best known solution.
$EstCost$	The estimated costs of the estimation.

Symbols used in Chapter 5

N	Set of companies
$p_{ij}(.)$	Pricing mechanism for company i for an order that has a location in the region of company j.
$\mathbf{p}(.)$	The vector which contains all pricing mechanisms.
ip	The fee for outsourcing an order.
ζ_j	Unit inter-depot costs from the hub to company j or the other way around.
m_i	Number of routes on an inter-depot line.

Symbols used in Chapters 6-8

N	$N = \{1, ..., n\}$ a finite set of players.
v	(N, v) a transferable utility game.
TU^N	The set of all transferable utility games with player set N.
$exc^P(v, S, x)$	The per-capita excess of coalition S for allocation x, $exc^P(v, S, x) = \frac{v(S) - \sum_{i \in S} c_i}{\lvert S \rvert}$.
(N, E, c)	A bankruptcy problem with player set N, estate E and claims c.
BR^N	The set of all bankruptcy problems with player set N.
$CEA(E, c)$	A bankruptcy rule which assigns the estate as equal as possible.
$CEL(E, c)$	A bankruptcy rule which assigns the estate such that the losses are as equal as possible.
$v_{E,c}$	The bankruptcy game, $v_{E,c}(S) = \max\{0, E - \sum_{i \in N \setminus S} c_i\}$.
$\sigma(E, c)$ and $\delta(c)$	The clights rule, see Definition 7.2.1 on 88.
$a_i(c)$	See Equation 7.3 on page 92.
$Q^\varepsilon(v, S, x)$	$Q^\varepsilon(v, S, x) = \frac{\sum_{i \in S} x_i}{v(S) + \varepsilon}$.

References

Archetti, C., N. Bianchessi, and M. G. Speranza (2013). Optimal solutions for routing problems with profits. *Discrete applied mathematics*, **161**, 547–557.

Archetti, C., D. Feillet, A. Hertz, and M. G. Speranza (2009). The capacitated team orienteering and profitable tour problems. *Journal of the Operational Research Society*, **60**, 831–842.

Archetti, C., A. Hertz, and M. G. Speranza (2007). Metaheuristics for the team orienteering problem. *Journal of Heuristics*, **13**(1), 49–76.

Archetti, C., M. G. Speranza, and D. Vigo (2014). Vehicle routing problems with profits. In: P. Toth and D. Vigo (Eds.), *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pp. 273–297. SIAM.

Audy, J.-F., S. D'amours, and L. M. Rousseau (2011). Cost allocation in the establishment of a collaborative transportation agreement- an application in the furniture industry. *Journal of the Operational Research Society*, **62**, 960–970.

Aumann, R. J. and M. Maschler (1985). Game theoretic analysis of a bankruptcy problem from the Talmud. *Journal of Economic Theory*, **36**, 195–213.

Baghuis, E. (2014). Outsourcing rule. Bachelor's Thesis, Tilburg University, Tilburg, the Netherlands.

Berger, S. and C. Bierwirth (2010). Solutions to the request reassignment problem in collaborative carrier networks. *Transportation Research Part E*, **46**, 627–638.

Bloos, M. and H. Kopfer (2009). Efficiency of transport collaboration mechanisms. *Communications of SIWN*, **6**, 23–28.

Bolduc, M.-C., J. Renoud, and F. Boctor (2007). A heuristic for the routing and carrier selection problem. *European Journal of Operational Research*, **183**, 926–932.

Bolduc, M.-C., J. Renoud, F. Boctor, and G. Laporte (2008). A perturbation metaheuristic for the vehicle routing problem with private fleet and common carriers. *Journal of the Operational Research Society*, **59**, 776–787.

Boussier, S., D. Feillet, and M. Gendrau (2007). An exact algorithm for team orienteering problems. *4OR: A quarterly Journal of Operations Research*, **5**, 211–230.

Christofides, N., A. Mingozzi, and P. Toth (1979). The vehicle routing problem. In: N. Christofides, A. Mingozzi, P. Toth, and C. Sandi (Eds.), *Combinatorial Optimization*, pp. 315–338. Wiley, Chichester.

Chu, C.-W. (2005). A heuristic algorithm for the truckload and less-than-truckload problem. *European Journal of Operational Research*, **165**, 657–667.

Côté, J.-F. and J.-Y. Potvin (2009). A tabu search heuristic for the vehicle routing problem with private fleet and common carrier. *European Journal of Operational Research*, **198**, 464–469.

Cruijssen, F., M. Cools, W. Dulleart, and H. Fleuren (2006). Emperisch onderzoek naar horizontale samenwerking (in Dutch). *Internal report, Tilburg University, Tilburg, the Netherlands*.

Cruijssen, F., W. Dulleart, and M. Cools (2007). Horizontal cooperation in logistics: Opportunities and impediments. *Transportation Research Part E: Logistics and Transportation Review*, **43**(2), 129–142.

Cruijssen, F., W. Dulleart, and H. Fleuren (2007). Horizontal cooperation in transport and logistics: A literature review. *Transportation Journal*, **46**(3), 22–39.

Daganzo, C. (1984). The distance traveled to visit N points with a maximum of C stops per vehicle: an analytic model and an application. *Transportation Science*, **18**(4), 331–350.

Dahl, S. and S. Derigs (2011). Cooperative planning in express carrier networks-An empirical study on the effectiveness of a real-time decision support system. *Decision Support Systems*, **51**(3), 620–626.

Dai, B. and H. Chen (2011). A multi-agent and auction-based framework and approach for carrier collaboration. *Logistics Research*, **3**(2–3), 101–120.

Dai, B. and H. Chen (2012). Profit allocation mechanisms for carrier collaboration in pickup and delivery service. *Computers & Industrial Engineering*, **62**(2), 633–643.

Faigle, U., W. Kern, S. P. Fekete, and W. Hochstätler (1998). The nucleon of cooperative games and an algorithm for matching games. *Mathematical Programming*, **83**, 195–211.

Feillet, D., P. Dejax, and M. Gendreau (2005). Traveling salesman problems with profits. *Transportation Science*, **39**(2), 188–205.

Figliozzi, M. (2009). Planning approximations to the average length of vehicle routing problems with time window constraints. *Transportation Research Part B*, **43**, 438–447.

Fleischmann, B. (1998). Design of freight traffic networks. In: B. Fleischmann, J. van Nunen, M. G. Speranza, and P. Stähly (Eds.), *Lecture Notes in Economics and Mathematical Systems: Advances in Distribution Logistics*, pp. 55–81. Springer-Verlag Berlin Heidelberg.

Fleischmann, B. (1999). Kooperation von herstellern in der konsumergüter-industrie (in German). In: J. Engelhard and B. J. Sinz (Eds.), *Kooperation im Wettbewerb*, pp. 167–186. Gabler.

Forrester, A. I. J., A. Sóbester, and A. J. Keane (2008). *Engineering Design via Surrogate Modelling*. Fourth Edition. John Wiley & Sons Ltd.

Frisk, M., M. Göthe-Lundgren, K. Jörnsten, and M. Rönnqvist (2010). Cost allocation in collaborative forest transportation. *European Journal of Operational Research*, **205**(2), 448–458.

Gillies, D. (1953). *Some Theorems on n-Person Games*. Ph. D. thesis, Princeton University Press, Princeton, New Jersey.

Golden, B. L., E. A. Wasil, J. P. Kelly, and I.-M. Chao (1998). The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: T. G. Crainic and G. Laporte (Eds.), *Fleet Management and Logistics*, pp. 33–56. Springer US.

Goudvis, P. (2001). Door de bomen het BOSS weer zien (in Dutch). Master's thesis, Tilburg University, Tilburg, the Netherlands.

Groote Schaarsberg, M., P. E. M. Borm, H. J. M. Hamers, and J. H. Reijnierse (2013). Game theoretic analysis of maximum cooperative purchasing situations. *Naval Research Logistics*, **8**(2013), 607–624.

Grotte, J. H. (1970). Computation of and Observations on the Nucleolus, the Normalized Nucleolus and the Central Games. Master's thesis, Cornell University, Ithaca, New York, the United States of America.

Gujo, O., M. Schwind, and J. Vykoukal (2007). The design of incentives in a combinatorial exchange for intra-enterprise logistics services. In: *9th international IEEE Conference on Enterprise Computing, E-Commerce, and E-Services, 2007, Tokyo, Japan*, pp. 443–446.

Hall, R. W. and M. Racer (1995). Transportation with common carrier and private fleets: system assignment and shipment frequency optimization. *IIE Transactions*, **27**(2), 217–225.

Hezarkhani, B., M. Slikker, and T. van Woensel (2015). A competitive solution for cooperative truckload delivery. *to appear in OR Spectrum*, 1–30.

Huijink, S., P. E. M. Borm, R. L. P. Hendrickx, and J. H. Reijnierse (2016). The proportionate nucleolus. *Preliminary manuscript,* Tilburg University, Tilburg, The Netherlands.

Huijink, S., P. E. M. Borm, J. Kleppe, and J. H. Reijnierse (2015). Bankruptcy and the per capita nucleolus: The claim-and-right rules family. *Mathematical Social Sciences*, **77**, 15–31.

Huijink, S., G. Kant, and M. J. P. Peeters (2014). An adaptable variable neighborhood search for the vehicle routing problem with order outsourcing. *Center Discussion Paper*, **2014-062**, 1–30.

Huijink, S., G. Kant, and M. J. P. Peeters (2015). Determining which orders to outsource in the vehicle routing problem with order outsourcing. *Center Discussion Paper*, **2015-049**, 1–17.

Kohlberg, E. (1971). On the nucleolus of a characteristic function game. *SIAM Journal of Applied Mathematics*, **20**, 62–66.

Krajewska, M. A. and H. Kopfer (2006). Collaborating freight forwarding enterprises. *OR Spectrum*, **28**(3), 301–317.

Krajewska, M. A., H. Kopfer, G. Laporte, S. Ropke, and G. Zaccour (2008). Horizontal cooperation among freight carriers: request allocation and profit sharing. *Journal of the Operational Research Society*, **59**, 1483–1491.

Kratica, J., T. Kostić, D. Tošić, D. Dugošija, and V. Filipović (2012). A genetic algorithm for the routing and carrier selection problem. *Computer Science and Information Systems*, **9**(1), 49–62.

Lui, P., Y. Wu, and N. Xu (2010). Allocating collaborative profit in less-than-truckload carrier alliance. *Journal of Service Science & Management*, **3**, 143–149.

Mackie-Mason, J. K. and H. R. Varian (1994). Generalized vickrey auctions. *Technical report, Department of Economics, University of Michigan, Michigan.*

Maschler, M., J. A. M. Potters, and S. H. Tijs (1992). The general nucleolus and the reduced game property. *International Journal of Game Theory*, **21**, 85–106.

Moon, I., J.-H. Lee, and J. Seong (2012). Vehicle routing problem with time windows considering overtime and outsourcing vehicles. *Expert Systems with Applications*, **39**(18), 13202–13213.

Moreno-Ternero, J. D. and A. Villar (2006). The TAL-family of rules for bankruptcy problems. *Social Choice and Welfare*, **27**(2), 231–249.

O'Neill, B. (1982). A problem of rights arbitration from the Talmud. *Mathematical Social Sciences*, **2**, 345–371.

Özener, O., O. Ergun, and M. Savelsbergh (2011). Lane−exchange mechanisms for truckload carrier collaboration. *Transportation Science*, **45**(1), 1–17.

Potvin, J.-Y. and M.-A. Naud (2011). Tabu search with ejection chains for the vehicle routing problem with private fleet and common carrier. *Journal of the Operational Research Society*, **62**, 326–336.

Schmeidler, D. (1969). The nucleolus of a characteristic function game. *SIAM Journal of Applied Mathematics*, **12**, 1163–1170.

Schönberger, J. (2005). *Operational Freight Carrier Planning.* Springer: Berlin, Heidelberg, New York.

Schwind, M., O. Gujo, and J. Vykoukal (2009). A combinatorial intra-enterprise exchange for logistics services. *Information Systems and E-Business Management*, **7**(4), 447–471.

Stenger, A., M. Schneider, and D. Goeke (2013). The prize-collecting vehicle routing problem with single and multiple depots and non-linear cost. *EURO Journal on Transportation and Logistics*, **2**, 57–87.

Stenger, A., D. Vigo, S. Enz, and M. Schwind (2013). An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. *Transportation Science*, **47**, 64–80.

Taillard, É. (1993). Taillard's VRP problems. http://mistic.heig-vd.ch/taillard/problemes.dir/vrp.dir/vrp.html. Accessed: December 3th, 2014.

Tang, L. and X. Wang (2006). Iterated local search algorithm based on very large−scale neighborhood for prize-collecting vehicle routing problem. *The International Journal of Advanced Manufacturing Technology*, **29**, 1246–1258.

Tang, L., X. Zhang, and Q. Guo (2009). Two hybrid metaheuristic algorithms for hot rolling scheduling. *ISIJ International*, **46**(4), 529–538.

Thomson, W. (2003). Axiomatic and game theoretic analysis of bankruptcy and taxation problems: a survey. *Mathematical Social Sciences*, **45**, 249–297.

Thomson, W. (2008). Two families of rules for the adjudication of conflicting claims. *Social Choice and Welfare*, **31**(4), 667–692.

Tjemkes, B., P. Vos, and K. Burgers (2012). *Strategic alliance management*. Routledge, London.

Verdonck, L., A. Caris, K. Ramaekers, and G. K. Janssens (2013). Collaborative logistics from the perspective of road transportation companies. *Transport Reviews*, **33**(6), 700–719.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, **16**(1), 8–37.

Vidal, T., T. G. Crainic, M. Gendreau, and C. Prins (2012). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *Cirrelt*, **2012-05**, 1–46.

Vidal, T., N. Maculan, L. S. Ochi, and P. H. V. Penna (2015). Large neighborhoods with implicit customer selection for vehicle routing problems with profits. *To appear in: Transportation Science*.

Vries, S. de and R. V. Vohra (2003). Combinatorial auctions: a survey. *The INFORMS Journal on Computing*, **15**(3), 284–309.

Wallmeier, E. (1983). *Der f-Nucleolus und ein dynamisches Verhandlungsmodell* *als Losungskonzepte fur Kooperative n-Personenspiele, Skripten zur Mathematischen Statistik 5*. Ph. D. thesis, Westfalische Wilhelms-Universitat Munster.

Wang, X. and H. Kopfer (2011). Increasing efficiency of freight carriers through collaborative transport planning: Chances and challenges. In: D. Ivanov, H. Kopfer, H.-D. Haasis, and J. Schönberger (Eds.), *Dynamics and Sustainability in International Logistics and Supply Chain Management - Proceedings of the 6th German-Russian Logistics SCM Workshop (DR-LOG 2011)*, pp. 41–50. Cuvillier Verlag, Göttingen.

Wang, X. and H. Kopfer (2014). Collaborative transportation planning of less-than-truckload freight. *OR Spectrum*, **36**, 357–380.

Wang, X., H. Kopfer, and M. Gendreau (2014). Operational transportation planning of freight forwarding companies in horizontal coalitions. *European Journal of Operational Research*, **237**, 1133–1141.

Samenwerking: Routering met Uitbesteding, Verdelingsproblemen en Nucleoni

Regionale vervoersbedrijven hebben vaak een depot of uitvalsbasis dicht bij de klanten die pakketten bezorgd willen hebben. Door deze ligging hoeven de chauffeurs niet ver te rijden om de pakketten op te halen. Helaas geldt dit niet voor het afleveren van de pakketten, aangezien de afleveradressen vaak verspreid zijn over het hele land (en soms zelfs erbuiten liggen). Voor de pakketten die een afleveradres ver weg hebben, moeten de chauffeurs ver rijden om er alleen al bij in de buurt te komen. Vaak is het ook nog eens zo dat er geen afleveradres van een ander pakket in de buurt ligt. Dit resulteert in hoge kosten waardoor het bedrijf geen rendabele 24-uurs levering kan garanderen. Wat kan een bedrijf doen om de kosten omlaag te krijgen van de pakketten die ver weg geleverd moeten worden?

De oplossing waar ik onderzoek naar heb gedaan is samenwerking met andere vervoersbedrijven, de zogeheten 'concullega's'. Er zijn verschillende manieren van samenwerken. Welke het beste is, hangt af van de eisen van het bedrijf aangezien elke vorm zijn eigen voor- en nadelen heeft. Drie vormen worden behandeld in Hoofdstuk 2. Twee daarvan, de centrale planning en het veilingmechanisme, hebben als nadeel dat niet bekend is hoe de winst verdeeld moet worden en dat er meer informatie gedeeld wordt dan de meeste bedrijven willen. Een samenwerking die deze twee nadelen niet heeft, is het vormen van een alliantie zoals TransMission, Netwerk Benelux, Teamtrans en Distri-XL doen. Een ander groot voordeel van het alliantieverband is de combinatie van onafhankelijkheid ten opzichte van de uit te besteden orders en de garantie dat de uitbestede orders ook echt afgenomen worden. Deze allianties bestaan uit verschillende regionale vervoersbedrijven, elk met een eigen regio. Doordat ze allemaal hetzelfde probleem hebben, namelijk dat het bezorgen van de pakketten ver weg relatief duur is, willen ze de pakketten uitbesteden naar elkaar. Om dit mogelijk te maken, spreken ze een prijsmechanisme af dat voor elk pakket de kosten aangeeft dat het bedrijf moet betalen aan het bedrijf dat het pakket gaat bezorgen. Het bepalen van het prijsmechanisme wordt behandeld in Hoofdstuk 5. In dit hoofdstuk creëer ik een nieuwe dataset. Op deze dataset bepaal ik het prijsmechanisme dat de totale kosten van de alliantie tracht te minimaliseren.

Gegeven de kosten voor het uitbesteden, moet elk bedrijf bepalen welke pakketten het uitbesteedt. Er zijn grofweg drie manieren om te bepalen welke pakketten het bedrijf het beste kan uitbesteden. De eerste manier is om alle mogelijkheden op een slimme manier af te gaan. Helaas is dat onmogelijk als er een realistisch aantal pakketten is. De tweede manier is door zogenoemde heuristieken te gebruiken. De heuristieken die ik hiervoor heb ontwikkeld, worden behandeld in Hoofdstuk 3. Deze nieuwe heuristieken blijken beter te zijn dan de heuristieken in de literatuur. De laatste manier is door middel van simpele rekenregels, bijvoorbeeld, door de kosten van het bezorgen te schatten en deze af te zetten tegen de uitbestedingskosten. De simpele rekenregels die ik heb ontwikkeld worden behandeld in Hoofdstuk 4. De nieuwe schattingen presteren het beste op zowel de oude als nieuwe datasets.

In deze alliantiesamenwerking wordt de winst automatisch verdeeld. Bij de twee andere manieren (centrale planning en veilingmechanisme) is een van de problemen hoe de kosten, of de extra winst, te verdelen. Een methode die de waarde van een bedrijf meet en aan de hand hiervan de winst verdeelt, is coöperatieve speltheorie. In Hoofdstuk 6 worden de verschillende manieren waarop de winst verdeeld kan worden voor onder andere transportbedrijven behandeld. Hoofdstuk 7 behandelt het curatorenprobleem. In dit probleem wil een curator de gedupeerden van een failliet bedrijf schadeloos stellen, maar er is niet genoeg geld voor iedereen. De vraag is: hoeveel krijgt elke gedupeerde? In dit hoofdstuk ontwikkel ik een nieuwe verdeelregel die samenvalt met de per capita nucleolus van het bijbehorende spel. Een nieuw allocatiemechanisme voor een algemeen coöperatief spel op basis van proportionaliteit wordt behandeld in Hoofdstuk 8.

Author index